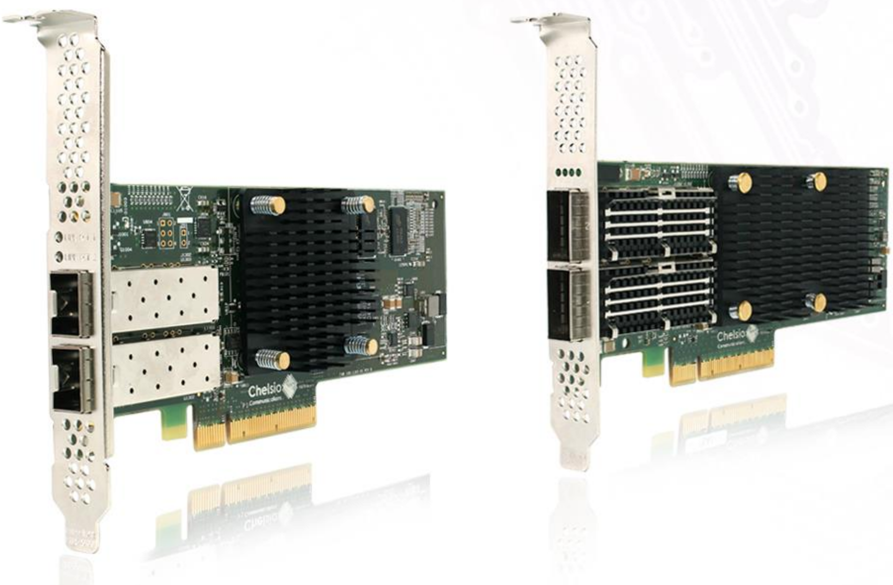
Chelsio Communications
Accelerate

Chelsio Unified Wire for Linux

Installation and User's Guide



Chelsio Communications (Headquarters)
209 North Fair Oaks Avenue,
Sunnyvale, CA 94085
U.S.A

www.chelsio.com

Tel: 408.962.3600
Fax: 408.962.3661

Chelsio (India) Private Limited
Subramanya Arcade, Floor 3, Tower B
No. 12, Bannerghatta Road,
Bangalore-560029
Karnataka,
India

Tel: +91-80-4039-6800

Chelsio KK (Japan)
Yamato Building 8F,
5-27-3 Sendagaya,
Shibuya-ku,
Tokyo 151-0051,
Japan

Sales
For all sales inquiries please send email to sales@chelsio.com

Support
For all support related questions please send email to support@chelsio.com

Document History

Version	Revision Date
1.0.0	12/08/2011
1.0.1	01/09/2013
1.0.2	01/27/2013
1.0.3	03/26/2013
1.0.4	04/12/2013
1.0.5	06/20/2013
1.0.6	08/17/2013
1.0.7	10/22/2013
1.0.8	03/08/2013
1.0.9	05/15/2013
1.1.0	07/26/2013
1.1.1	08/14/2013
1.1.2	12/06/2013
1.1.3	12/19/2013
1.1.4	03/13/2014
1.1.5	05/02/2014
1.1.6	06/30/2014
1.1.7	10/22/2014
1.1.8	11/04/2014
1.1.9	02/05/2015
1.2.0	03/04/2015
1.2.1	03/25/2015
1.2.2	06/03/2015
1.2.3	08/05/2015
1.2.4	02/29/2016
1.2.5	04/27/2016
1.2.6	08/25/2016
1.2.7	10/07/2016
1.2.8	10/18/2016
1.2.9	11/11/2016
1.3.0	12/05/2016

TABLE OF CONTENTS

I. Chelsio Unified Wire

1. Introduction

Thank you for choosing Chelsio Unified Wire adapters. These high speed, single chip, single firmware cards provide enterprises and data centers with high performance solutions for various Network and Storage related requirements.

The **Terminator** series is Chelsio's next generation of highly integrated, hyper-virtualized 1/10/25/40/50/100GbE controllers. The adapters are built around a programmable protocol-processing engine, with full offload of a complete Unified Wire solution comprising NIC, TOE, iWARP RDMA, iSCSI, FCoE and NAT support. It scales to true 100Gb line rate operation from a single TCP connection to thousands of connections, and allows simultaneous low latency and high bandwidth operation thanks to multiple physical channels through the ASIC.

Ideal for all data, storage and high performance clustering applications, the Unified Wire adapters enable a unified fabric over a single wire by simultaneously running all unmodified IP sockets, Fibre Channel and InfiniBand applications over Ethernet at line rate.

Designed for deployment in virtualized data centers, cloud service installations and high performance computing environments, Chelsio adapters bring a new level of performance metrics and functional capabilities to the computer networking industry.

Chelsio Unified Wire software comes in two formats: Source code and RPM package forms. Installing from source requires compiling the package to generate the necessary binaries. You can choose this method when you are using a custom-built kernel. You can also install the package using the interactive GUI installer. In other cases, download the RPM package specific to your operating system and follow the steps mentioned to install the package. Please note that the OFED software required to install Chelsio iWARP driver comes bundled in both source as well as RPM packages.

This document describes the installation, use and maintenance of the software and its various components.

1.1. Features

The Chelsio Unified Wire Package uses a single command to install various drivers and utilities. It consists of the following software:

- **Network (NIC/TOE)**
- **Virtual Function Network (vNIC)**
- **iWARP (RDMA)**
- **iSER**
- **WD-UDP**
- **NVMe-oF**
- **LIO iSCSI Target Offload Driver**

- **iSCSI PDU Offload Target**
- **iSCSI PDU Offload Initiator**
- **Crypto Offload**
- **Data Center Bridiging (DCB)**
- **FCoE full offload Initiator**
- **Offload Bonding driver**
- **Offload Multi-adapter Failover(MAFO)**
- **UDP Segmentation Offload and Pacing**
- **Offload IPv6 driver**
- **Classification and Filtering feature**
- **Traffic Management feature (TM)**
- **Unified Wire Manager (UM)**
- **Unified Boot Software**
- **Utility Tools (cop, cxgbtool, t4_perftune, benchmark tools, sniffer & tracer)**
- **libs (iWARP and WD-UDP libraries)**

For detailed instructions on loading, unloading and configuring the drivers/tools please refer to their respective sections.

1.2. Hardware Requirements

The Chelsio Unified Wire software supports Chelsio Terminator series of Unified Wire adapters. To know more about the list of adapters supported by each driver, please refer to their respective sections.

1.3. Software Requirements

The Chelsio Unified Wire software has been developed to run on 64-bit Linux based platforms and therefore it is a base requirement for running the driver. To know more about the complete list of operating systems supported by each driver, please refer to their respective sections.

1.4. Package Contents

1.4.1. Source Package

The Chelsio Unified Wire source package consists of the following files/directories:

- **debrules**: This directory contains packaging specification files required for building Debian packages.
- **docs:** This directory contains support documents - README, Release Notes and User's Guide (this document) for the software.

- **libs:** This directory is for libraries required to install the WD-UDP and iWARP drivers. The libibverbs library has implementation of RDMA verbs which will be used by iWARP applications for data transfers. The librdmacm library works as an RDMA connection manager. The libcxgb4 library works as an interface between the above mentioned generic libraries and Chelsio iWARP driver. The libcxgb4_sock library is a LD_PRELOAD-able library that accelerates UDP Socket communications transparently and without recompilation of the user application.
- **OFED**: This directory contains supported OFED packages.
- **RPM-Manager**: This directory contains support scripts used for cluster deployment.
- **scripts:** Support scripts used by the Unified Wire Installer.
- **specs:** The packaging specification files required for building RPM packages.
- **src:** Source code for different drivers.
- **support:** This directory contains source files for the dialog utility.
- **tools:**
    - **autoconf-x.xx**: This directory contains the source for Autoconf tool needed for WD-UDP and iWARP libraries.
    - **benchmarks:** This directory contains various benchmarking tools to measure throughput and latency of various networks.
    - **chelsio_adapter_config**: This directory contains scripts and binaries needed to configure Chelsio 40G adapters.
    - **cop:** The cop tool compiles offload policies into a simple program form that can be loaded into the kernel and interpreted. These offload policies are used to determine the settings to be used for various connections. The connections to which the settings are applied are based on matching filter specifications. Please find more details on this tool in its manual page (run `man cop` command).
    - **cudbg:** Chelsio Unified Debug tool which facilitates collection and viewing of various debug entities like register dump, Devlog, CIM LA, etc.
    - **cxgbtool:** The cxgbtool queries or sets various aspects of Chelsio network interface cards. It complements standard tools used to configure network settings and provides functionality not available through such tools. Please find more details on this tool in its manual page (run `man cxgbtool` command).

      Note *To use cxbtool for FCoE Initiator driver, use [root@host~]# cxgbtool stor -h*

    - **rdma_tools:** This directory contains iWARP benchmarking tools.
    - **t4_sniffer:** This directory contains sniffer tracing and filtering libraries. See **WD Sniffing and Tracing** chapter for more information.
    - **um:** This directory contains Unified Wire Manager RPMs for different distributions and Management Station configuration files.
    - **90-rdma.rules:** This file contains udev rules needed for running RDMA applications as a non-root user.
    - **chdebug:** This script collects operating system environment details and debug information which can be sent to the support team, to troubleshoot Chelsio hardware/software related issues.

- **chiscsi_set_affinity.sh**: This shell script is used for mapping iSCSI Worker threads to different CPUs.
- **chsetup:** The chsetup tool loads NIC, TOE and iWARP drivers, and creates WD-UDP configuration file.
- **chstatus:** This utility provides status information on any Chelsio NIC in the system.
- **t4_latencytune.sh**: Script used for latency tuning of Chelsio adapters.
- **t4_perftune.sh:** This shell script is to tune the system for higher performance. It achieves it through modifying the IRQ-CPU binding. This script can also be used to change Tx coalescing settings.
- **t4-forward.sh:** RFC2544 Forward test tuning script.
- **uname_r:** This file is used by *chstatus* script to verify if the Linux platform is supported or not.
- **wdload:** UDP acceleration tool.
- **wdunload:** Used to unload all the loaded Chelsio drivers.
- **target**: The directory contains *targetcli* installation files and dependent components.

- **install.py, dialog.py:** Python scripts needed for the GUI installer.
- **EULA:** Chelsio's End User License Agreement
- **install.log:** File containing installation summary.
- **Makefile:** The Makefile for building and installing from the source.
- **sample_machinefile:** Sample file used during iWARP installation on cluster nodes.
- **Uboot:** There are two sub-directories in the *Uboot directory: OptionROM* and *LinuxDUD.* The *OptionROM* directory contains Unified Boot Option ROM image (*cubt4.bin*), uEFI driver (*ChelsioUD.efi*), default boot configuration file (*bootcfg*) and a legacy flash utility (*cfut4.exe*), which can be used to flash the option ROM onto Chelsio adapters (CNAs).
  The *LinuxDUD* directory contains image (.img) files required to update drivers for Linux distributions.

1.4.2. RPM package

The Chelsio Unified Wire RPM package consists of the following:

- **config**: This directory contains firmware configuration files.
- **docs:** This directory contains support documents i.e. README, Release Notes and User's Guide (this document) for the software.
- **DRIVER-RPMS:** RPM packages of Chelsio drivers.
- **OFED-RPMS:** OFED RPM packages required to install iWARP driver.
- **scripts:** Support scripts used by the Unified Wire Installer.
- **EULA:** Chelsio's End User License Agreement.
- **install.py:** Python script that installs the RPM package. See **Software/Driver Installation** section for more information.
- **uninstall.py:** Python script that uninstalls the RPM package. See **Software/Driver Uninstallation** section for more information.
- **Uboot:** There are two sub-directories in the *Uboot directory: OptionROM* and *LinuxDUD.* The *OptionROM* directory contains Unified Boot Option ROM image (*cubt4.bin*), uEFI driver

(*ChelsioUD.efi*), default boot configuration file (*bootcfg*) and a legacy flash utility (*cfut4.exe*), which can be used to flash the option ROM onto Chelsio adapters (CNAs).
The *LinuxDUD* directory contains image (.img) files required to update drivers for Linux distributions.

2. Hardware Installation

Follow these steps to install Chelsio adapter in your system:

1. Shutdown/power off your system.
2. Power off all remaining peripherals attached to your system.
3. Unpack the Chelsio adapter and place it on an anti-static surface.
4. Remove the system case cover according to the system manufacturer's instructions.
5. Remove the PCI filler plate from the slot where you will install the Ethernet adapter.
6. For maximum performance, it is highly recommended to install the adapter into a PCIe x8/x16 slot.
7. Holding the Chelsio adapter by the edges, align the edge connector with the PCI connector on the motherboard. Apply even pressure on both edges until the card is firmly seated. It may be necessary to remove the SFP (transceiver) modules prior to inserting the adapter.
8. Secure the Chelsio adapter with a screw, or other securing mechanism, as described by the system manufacturer's instructions. Replace the case cover.
9. After securing the card, ensure that the card is still fully seated in the PCIE x8/x16 slot as sometimes the process of securing the card causes the card to become unseated.
10. Connect a fiber/twinax cable, multi-mode for short range (SR) optics or single-mode for long range (LR) optics, to the Ethernet adapter or regular Ethernet cable for the 1Gb Ethernet adapter.
11. Power on your system.
12. Run *update-pciids* command to download the current version of PCI ID list

```
[root@host~]# update-pciids
%   Total   %  Received  %  Xferd  Average Speed  Time   Time   Time Current
                                   Dload  Upload Total  Spent Left  Speed
100 198k  100  198k     0    0     491k   0 --:--:-- --:--:-- --:--:-- 626k
Done.
```

13. Verify if the adapter was installed successfully by using the *lspci* command

```
[root@host~]# lspci |grep -i Chelsio
07:00.0 Ethernet controller: Chelsio Communications Inc T520-LL-CR Unified
Wire Ethernet Controller
07:00.1 Ethernet controller: Chelsio Communications Inc T520-LL-CR Unified
Wire Ethernet Controller
07:00.2 Ethernet controller: Chelsio Communications Inc T520-LL-CR Unified
Wire Ethernet Controller
07:00.3 Ethernet controller: Chelsio Communications Inc T520-LL-CR Unified
Wire Ethernet Controller
07:00.4 Ethernet controller: Chelsio Communications Inc T520-LL-CR Unified
Wire Ethernet Controller
```

```
07:00.5 SCSI storage controller: Chelsio Communications Inc T520-LL-CR
Unified Wire Storage Controller
07:00.6 Fibre Channel: Chelsio Communications Inc T520-LL-CR Unified Wire
Storage Controller
```

For Chelsio adapters, the physical functions are currently assigned as:

- Physical functions 0 - 3: for the SR-IOV functions of the adapter
- Physical function 4: for all NIC functions of the adapter
- Physical function 5: for iSCSI
- Physical function 6: for FCoE
- Physical function 7: Currently not assigned

Once Unified Wire package is installed and loaded, examine the output of `dmesg` to see if the card is discovered.

```
eth2: Chelsio T520-LL rev 1 1000/10GBASE-SFP RNIC MSI-X, Offload capable
0000:07:00.4: S/N: RE12130097, P/N: 11011675004
```

The above outputs indicate the hardware configuration of the adapters as well as the Serial numbers.

Note *Network device names for Chelsio's physical ports are assigned using the following convention: the port farthest from the motherboard will appear as the first network interface. However, for T5 40G and T420-BT adapters, the association of physical Ethernet ports and their corresponding network device names is opposite. For these adapters, the port nearest to the motherboard will appear as the first network interface.*

3. Software/Driver Installation

There are two main methods to install the Chelsio Unified Wire package: from source and RPM.
If you decide to use source, you can install the package using CLI or GUI mode.
If you decide to use RPM, you can install the package using Menu or CLI mode.

Irrespective of the method chosen for installation, the machine needs to be rebooted for changes to take effect.

The following table describes the various *configuration tuning options* available during installation and drivers/software installed with each option by default:

Configuration Tuning Option	Description	Driver/Software installed
Unified Wire	Configures adapters to run multiple protocols like NIC/TOE, iWARP, iSCSI and FCoE Initiator simultaneously.	NIC/TOE, vNIC, iWARP, iSER, WD-UDP, NVMe-oF, LIO iSCSI Target, iSCSI Target, iSCSI Initiator, Crypto, DCB, FCoE Initiator, Bonding, MAFO, IPv6, Sniffer & Tracer, UM (Agent, Client, WebGUI), Filtering, TM
Low latency Networking*	Configures adapters to run NIC/TOE and iWARP traffic with low latency specially needed for financial applications.	NIC/TOE, iWARP, WD-UDP, IPv6, Sniffer & Tracer, Bonding, MAFO, UM (Agent, Client, WebGUI), Filtering, TM
High capacity RDMA*	Configures adapters to establish a large number of RDMA connections.	NIC/TOE, iWARP, WD-UDP, Bonding, MAFO, IPv6, Sniffer & Tracer, UM (Agent, Client, WebGUI), Filtering, TM
RDMA Performance*	Improves RDMA performance.	NIC/TOE, iWARP, UM (Agent, Client, WebGUI)
High capacity TOE*	Configures adapters to establish a large number of TOE connections.	NIC/TOE, Bonding, MAFO, IPv6, UM (Agent, Client, WebGUI), Filtering, TM
iSCSI Performance+	Improves iSCSI performance.	NIC/TOE, LIO iSCSI Target, iSCSI Target, iSCSI Initiator, Bonding, DCB, UM (Agent, Client, WebGUI)
UDP Seg.Offload & Pacing*	Configures adapters to establish a large number of UDP Segmentation Offload connections.	NIC/TOE, IPv6, USO, Bonding, UM (Agent, Client, WebGUI), Filtering, TM
Wire Direct Latency+	Configures adapters to provide low Wire Direct latency.	NIC/TOE, iWARP, WD-UDP, UM (Agent, Client, WebGUI)
High Capacity WD*	Configures adapters to establish a large number of WD-UDP connections.	NIC/TOE, WD-UDP, UM (Agent, Client, WebGUI)
Hash Filter#	Configures adapters to create more filters.	NIC, Filtering, UM (Agent, Client, WebGUI)
Memory Free+ ^	Configures adapters in a memory-free configuration supporting offload protocols.	NIC/TOE, iWARP, UM (Agent, Client, WebGUI)

* Supported on T4/T5

+ Supported only on T5

Supported on T5/T6

^ Beta release. Should be used only with SO adapters.

3.1. Pre-requisites

Depending on the component you choose to install, please ensure that the following requirements are met, before proceeding with the installation.

- If you want to install OFED with NFS-RDMA support, please refer “Setting up NFS-RDMA” in **iWARP (RDMA)** (Click here).
- If you’re planning to install iSCSI PDU Offload Initiator, please install openssl-devel package.
- IPv6 should be enabled in the machine to use the RPM Packages.
- To install NVMe-oF driver:
    - Python v2.7 or above is required. If Python v2.7 is not already present in the system, or if an older version exists, v2.7.10 provided in the package will be installed.
    - *openssl-devel*, *bzip2-devel*, *zlib-devel*, *ncurses-devel*, *sqlite-devel* packages should be installed in the machine.
    - Enable the following parameters in the kernel configuration file:

```
CONFIG_BLK_DEV_NVME=m
CONFIG_NVME_RDMA=m
CONFIG_NVME_TARGET=m
CONFIG_NVME_TARGET_RDMA=m
CONFIG_NVME_RDMA=m
CONFIG_BLK_DEV_NULL_BLK=m
CONFIG_CONFIGFS_FS=y
```

- To use the crypto offload feature, compile the kernel with the following parmeters set as:

```
CONFIG_KEYS=y
CONFIG_KEYS_DEBUG_PROC_KEYS=y
CONFIG_SECURITY=y
CONFIG_SECURITY_NETWORK=y
CONFIG_SECURITY_NETWORK_XFRM=y
CONFIG_LSM_MMAP_MIN_ADDR=65536
CONFIG_SECURITY_SELINUX=y
CONFIG_SECURITY_SELINUX_BOOTPARAM=y
CONFIG_SECURITY_SELINUX_BOOTPARAM_VALUE=1
CONFIG_SECURITY_SELINUX_DISABLE=y
CONFIG_SECURITY_SELINUX_DEVELOP=y
CONFIG_SECURITY_SELINUX_AVC_STATS=y
CONFIG_SECURITY_SELINUX_CHECKREQPROT_VALUE=1
CONFIG_DEFAULT_SECURITY_SELINUX=y
CONFIG_DEFAULT_SECURITY="selinux"
CONFIG_CRYPTO=y
CONFIG_CRYPTO_FIPS=y
CONFIG_CRYPTO_ALGAPI=y
CONFIG_CRYPTO_ALGAPI2=y
CONFIG_CRYPTO_AEAD=y
CONFIG_CRYPTO_AEAD2=y
```

```
CONFIG_CRYPTO_BLKCIPHER=y
CONFIG_CRYPTO_BLKCIPHER2=y
CONFIG_CRYPTO_HASH=y
CONFIG_CRYPTO_HASH2=y
CONFIG_CRYPTO_RNG=y
CONFIG_CRYPTO_RNG2=y
CONFIG_CRYPTO_PCOMP=y
CONFIG_CRYPTO_PCOMP2=y
CONFIG_CRYPTO_MANAGER=y
CONFIG_CRYPTO_MANAGER2=y
CONFIG_CRYPTO_NULL=y
CONFIG_CRYPTO_WORKQUEUE=y
CONFIG_CRYPTO_CRYPTD=y
CONFIG_CRYPTO_AUTHENC=y
CONFIG_CRYPTO_TEST=m
CONFIG_CRYPTO_CCM=y
CONFIG_CRYPTO_GCM=y
CONFIG_CRYPTO_SEQIV=y
CONFIG_CRYPTO_CBC=y
CONFIG_CRYPTO_CTR=y
CONFIG_CRYPTO_CTS=y
CONFIG_CRYPTO_ECB=y
CONFIG_CRYPTO_XTS=y
CONFIG_CRYPTO_HMAC=y
CONFIG_CRYPTO_GHASH=y
CONFIG_CRYPTO_MD4=m
CONFIG_CRYPTO_MD5=y
CONFIG_CRYPTO_SHA1=y
CONFIG_CRYPTO_SHA256=y
CONFIG_CRYPTO_SHA512=y
CONFIG_CRYPTO_AES=y
CONFIG_CRYPTO_AES_X86_64=y
CONFIG_CRYPTO_DEFLATE=y
CONFIG_CRYPTO_ZLIB=y
CONFIG_CRYPTO_LZO=y
CONFIG_CRYPTO_ANSI_CPRNG=y
CONFIG_CRYPTO_USER_API=y
CONFIG_CRYPTO_USER_API_HASH=y
CONFIG_CRYPTO_USER_API_SKCIPHER=y
CONFIG_CRYPTO_HW=y
```

3.2. Enabling RDMA on ARM Platforms

RDMA is disabled by default in RHEL 7.X build of ARM architecture. To enable this feature, follow the steps mentioned below:

i. Download the kernel source package and extract it.
ii. Create a kernel configuration file.

```
[root@host~]# make oldconfig
```

iii. The above command will create a configuration file *.config* in the same location. Edit the file and enable the following parameters as follows:

```
CONFIG_NET_VENDOR_CHELSIO=y
CONFIG_INFINIBAND=y
```

iv. Compile the kernel.
v. During kernel compilation, please ensure that the following parameters are set as follows:

```
CONFIG_CHELSIO_T1=m
CONFIG_CHELSIO_T1_1G=y
CONFIG_CHELSIO_T3=m
CONFIG_CHELSIO_T4=m
CONFIG_CHELSIO_T4VF=m
CONFIG_INFINIBAND_USER_MAD=m
CONFIG_INFINIBAND_USER_ACCESS=m
CONFIG_INFINIBAND_USER_MEM=y
CONFIG_INFINIBAND_CXGB3=m
CONFIG_INFINIBAND_CXGB3_DEBUG=y
CONFIG_INFINIBAND_CXGB4=m
CONFIG_SCSI_CXGB3_ISCSI=m
CONFIG_SCSI_CXGB4_ISCSI=m
```

vi. Install the kernel.
vii. Reboot into the newly installed kernel.

3.3. Allowing unsupported modules on SLES

On SLES11 SPx platforms, edit the */etc/modprobe.d/unsupported-modules* file and change *allow_unsupported_modules* to 1.

On SLES12 SPx platforms, edit the */etc/modprobe.d/10-unsupported-modules.conf* file and change *allow_unsupported_modules* to 1.

3.4. Installing Chelsio Unified Wire from source

3.4.1. GUI mode (with Dialog utility)

i. Download the Unified Wire driver package (tarball) from Chelsio Download Center, http://service.chelsio.com/

ii. Untar the tarball using the following command:

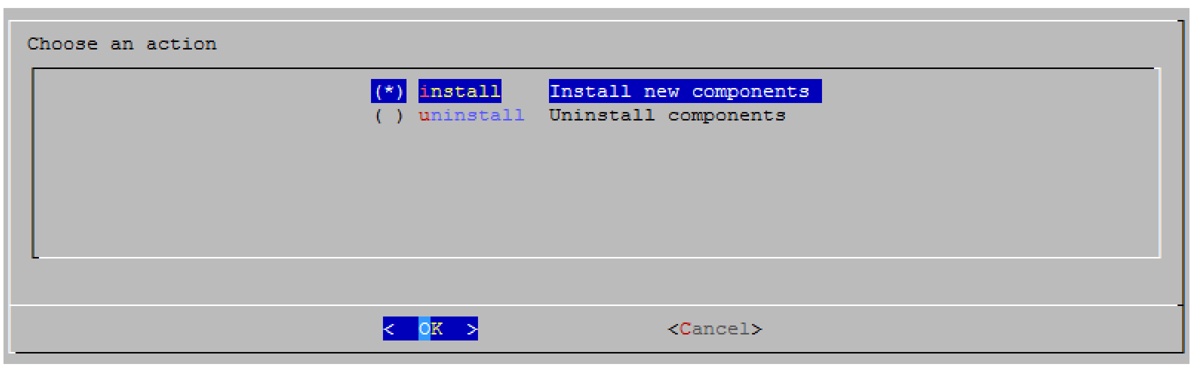
```
[root@host~]# tar zxvfm <driver_package>.tar.gz
```

iii. Change your current working directory to Chelsio Unified Wire package directory and run the following script to start the GUI installer:

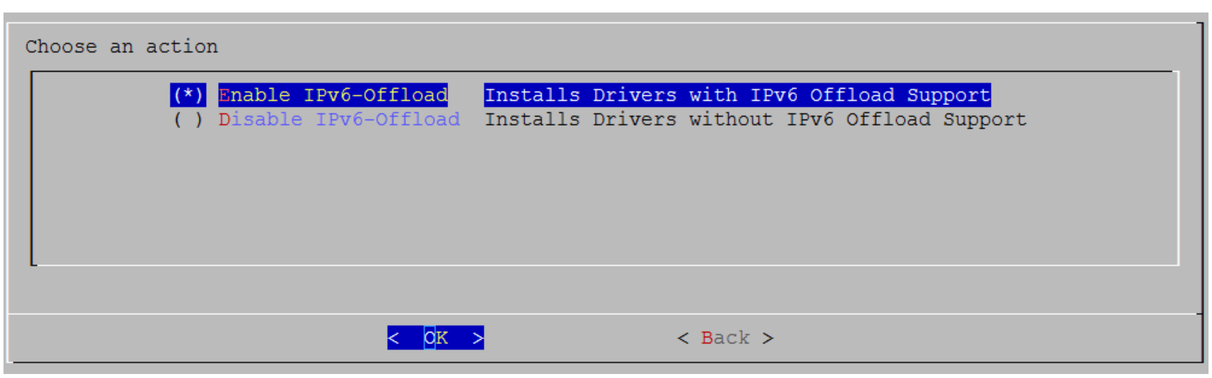
```
[root@host~]# ./install.py
```

iv. If **Dialog** utility is present, you can skip to step (v). If not, press 'y' to install it when the installer prompts for input.

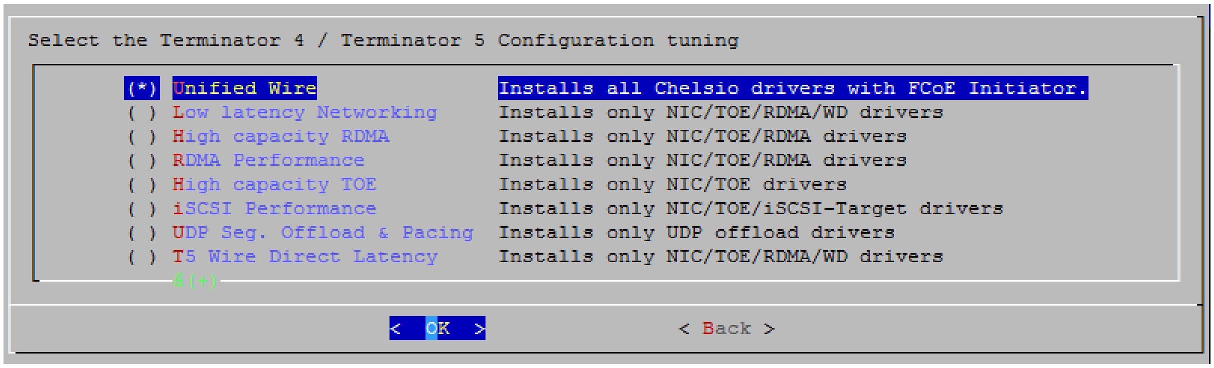
v. Select "install" under "Choose an action"



vi. Select *Enable IPv6-Offload* to install drivers with IPv6 Offload support or *Disable IPv6-offload* to continue installation without IPv6 offload support.

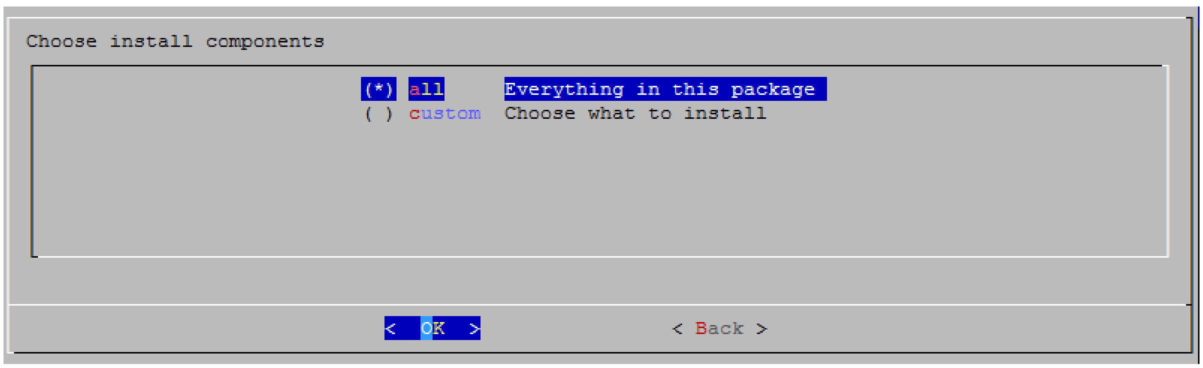


vii. Select the required configuration tuning option:



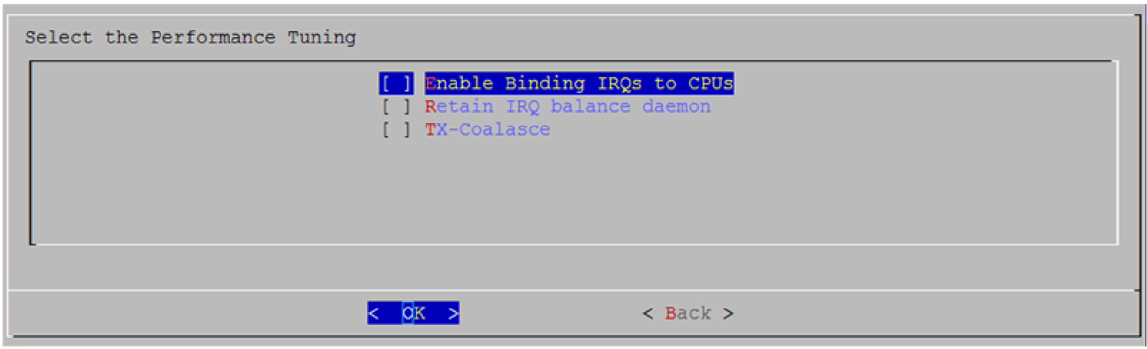
Note *The tuning options may vary depending on the Linux distribution.*

viii. Under “Choose install components”, select “all” to install all the related components for the option chosen in step (vii) or select “custom” to install specific components.



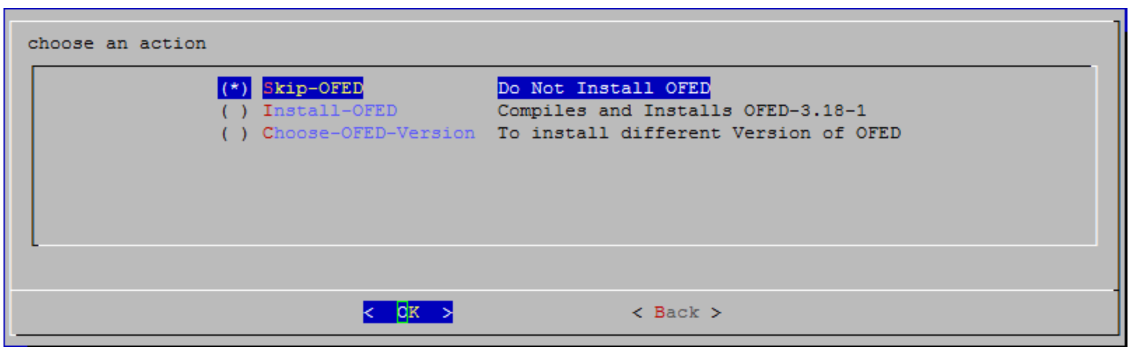
Important *To install benchmark tools, please select “custom option”.*

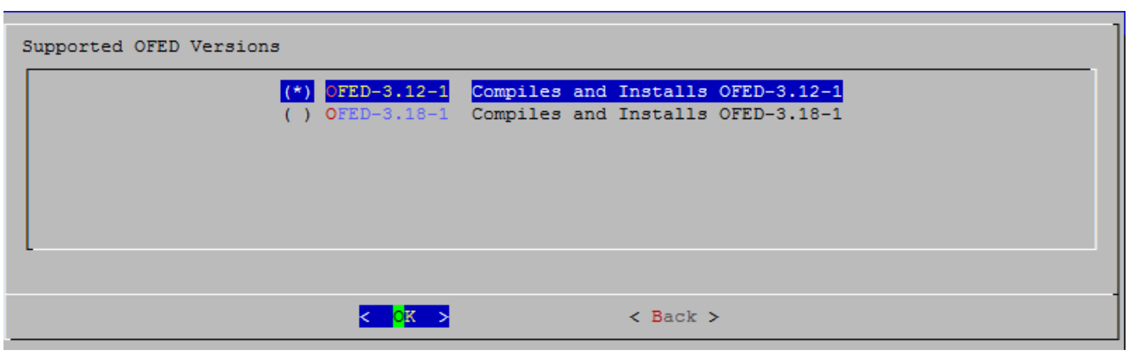
ix. Select the required performance tuning option.
   a. Enable Binding IRQs to CPUs: Bind MSI-X interrupts to different CPUs and disable IRQ balance daemon.
   b. *Retain IRQ balance daemon*: Do not disable IRQ balance daemon.
   c. *TX-Coalasce*: Write tx_coal=2 to modprobe.d/conf.



Note *For more information on the Performance tuning options, please refer to* Performance Tuning *section of the* ***Network (NIC/TOE)*** *chapter.*

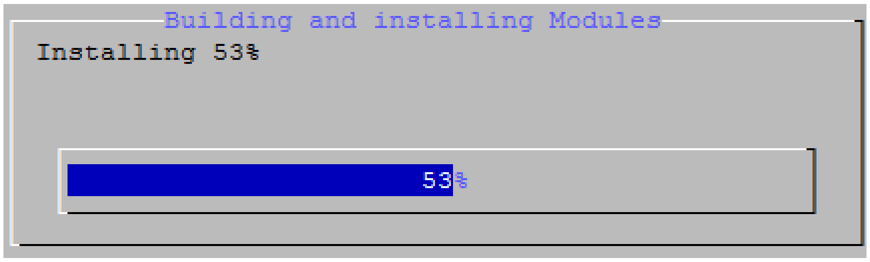
x. If you already have the required version of OFED software installed, you can skip this step by selecting *Skip-OFED*.
To install OFED-3.18-2 choose the *Install-OFED* option. To install OFED-3.12-1, select *Choose-OFED-Version* and then *OFED-3.12-1*.



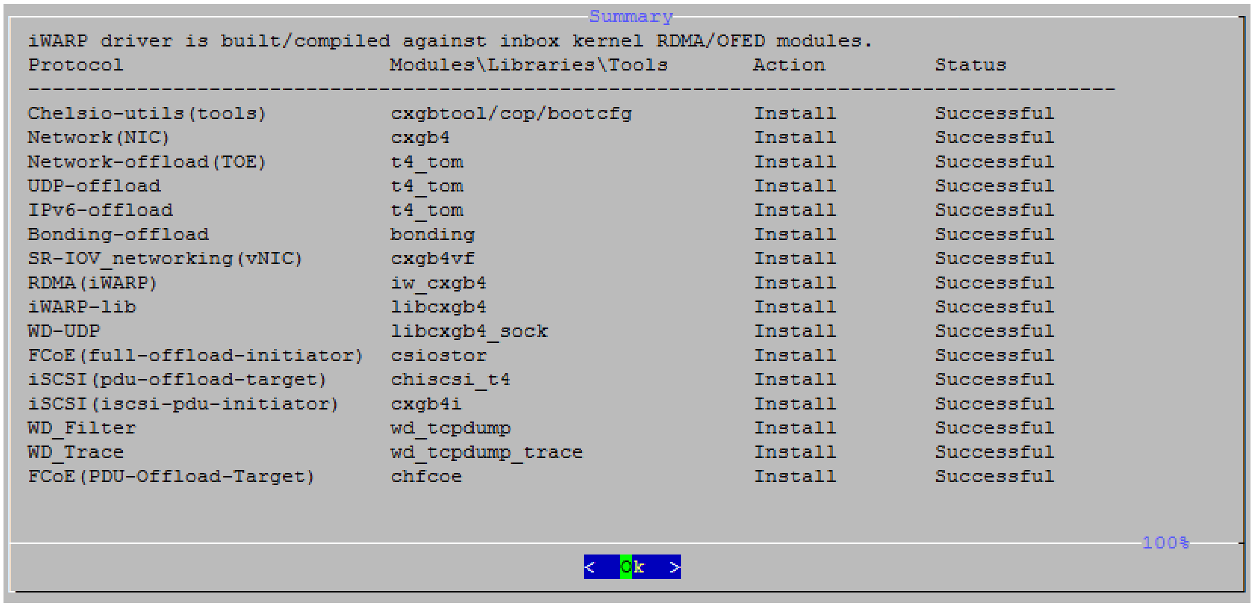


Note *This step will be prompted only for OFED supported platforms.*

xi. The selected components will now be installed:



xii. After successful installation, summary of installed components will be displayed.


Summary

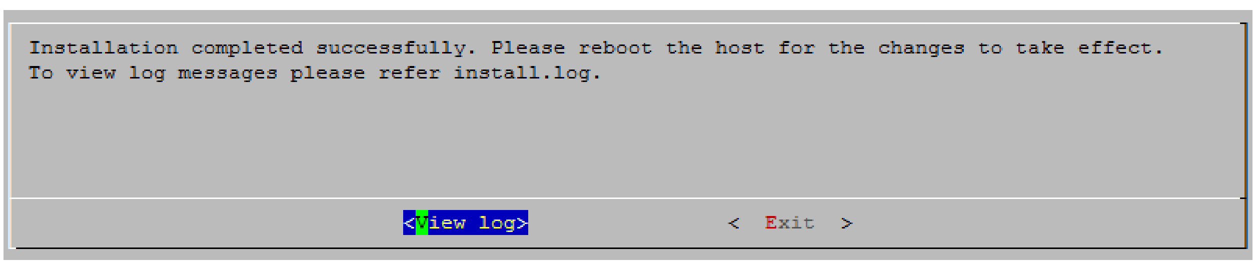
iWARP driver is built/compiled against inbox kernel RDMA/OFED modules.

Protocol	Modules\Libraries\Tools	Action	Status
Chelsio-utils(tools)	cxgbtool/cop/bootcfg	Install	Successful
Network(NIC)	cxgb4	Install	Successful
Network-offload(TOE)	t4_tom	Install	Successful
UDP-offload	t4_tom	Install	Successful
IPv6-offload	t4_tom	Install	Successful
Bonding-offload	bonding	Install	Successful
SR-IOV_networking(vNIC)	cxgb4vf	Install	Successful
RDMA(iWARP)	iw_cxgb4	Install	Successful
iWARP-lib	libcxgb4	Install	Successful
WD-UDP	libcxgb4_sock	Install	Successful
FCoE(full-offload-initiator)	csiostor	Install	Successful
iSCSI(pdu-offload-target)	chiscsi_t4	Install	Successful
iSCSI(iscsi-pdu-initiator)	cxgb4i	Install	Successful
WD_Filter	wd_tcpdump	Install	Successful
WD_Trace	wd_tcpdump_trace	Install	Successful
FCoE(PDU-Offload-Target)	chfcoe	Install	Successful

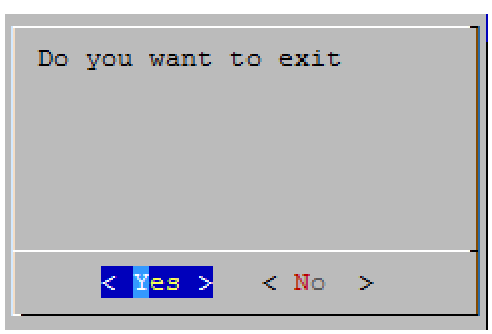
100%

< Ok >

xiii. Select “View log” to view the installation log or “Exit” to continue.



xiv. Select “Yes” to exit the installer or “No” to go back.



xv. Reboot your machine for changes to take effect.

Note *Press Esc or Ctrl+C to exit the installer at any point of time.*

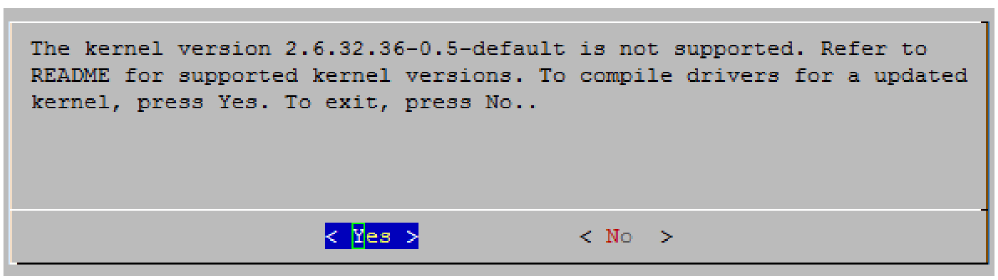
3.4.2.1. Installation on updated kernels

If the kernel version on your Linux distribution is updated, follow the steps mentioned below to install the Unified Wire package:

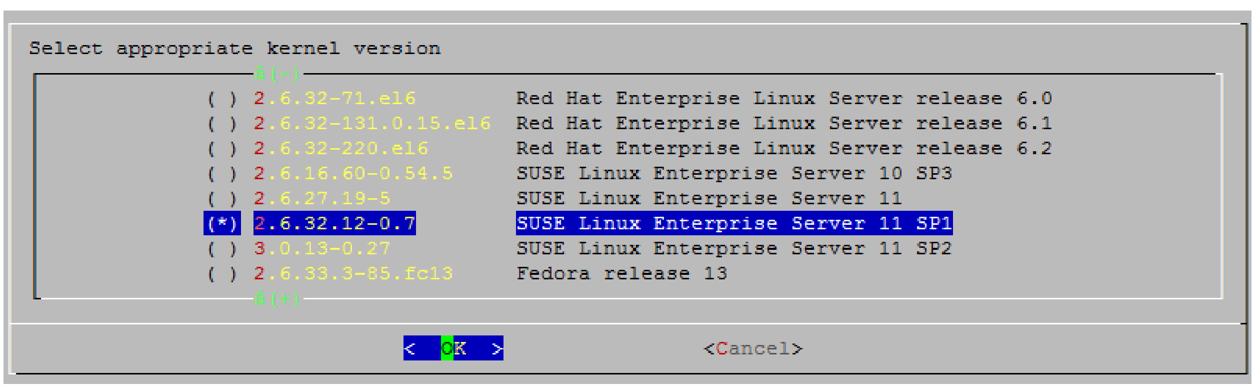
i. Change your current working directory to Chelsio Unified Wire package directory and run the following script to start the GUI installer:

```
[root@host~]# ./install.py
```

ii. Select "Yes" to continue with the installation on the updated kernel or "No" to exit.



iii. Select the nearest supported kernel version from the list and select "OK".



iv. Follow steps (v) to (xv) mentioned in the previous section.

3.4.3. CLI mode (without Dialog utility)

If your system does not have **Dialog** or you choose not to install it, follow the steps mentioned below to install the Unified Wire package:

i. Download the Unified Wire driver package from Chelsio Download Center, http://service.chelsio.com/
ii. Untar the tarball using the following command:

```
[root@host~]# tar zxvfm <driver_package>.tar.gz
```

iii. Change your current working directory to Chelsio Unified Wire package directory and run the following script to start the installer:

```
[root@host~]# ./install.py
```

iv. When the installer prompts you for your input, press ‘n’ to continue installation without the **Dialog** utility.
v. Enter the number corresponding to the Configuration tuning option in the Input field and press Enter.
vi. If you already have the required version of OFED software installed, you can skip this step. To install OFED-3.18-2 choose the *Install-OFED* option. To skip this step, select *Skip-OFED*.

Note *This step will be prompted only for OFED supported platforms.*

vii. The selected components will now be installed.
After successful installation you can press 1 to view the installation log. Press any other key to exit from the installer.

Important *To customize the installation, view the help by typing* `[root@host~]#./install.py -h`

viii. Reboot your machine for changes to take effect.

3.4.3.1. iWARP driver installation on Cluster nodes

Important *Please make sure that you have enabled password less authentication with ssh on the peer nodes for this feature to work.*

Chelsio’s Unified Wire package allows installing iWARP drivers on multiple Cluster nodes with a single command. Follow the procedure mentioned below:

i. Create a file (*machinefilename*) containing the IP addresses or hostnames of the nodes in the cluster. You can view the sample file, *sample_machinefile,* provided in the package to view the format in which the nodes have to be listed.

ii. Now, execute the following command:

```
[root@host~]# ./install.py -C  -m <machinefilename>
```

iii. Select the required configuration tuning option. The tuning options may vary depending on the Linux distribution.

iv. Select the required Cluster Configuration.

v. If you already have the required version of OFED software installed, you can skip this step. To install OFED-3.18-2 choose the *Install-OFED* option. To skip this step, select *Skip-OFED*.

vi. The selected components will now be installed.

The above commands will install iWARP (*iw_cxgb4*) and TOE (*t4_tom*) drivers on all the nodes listed in the *machinefilename* file.

3.4.4. CLI mode

i. Download the Unified Wire driver package from Chelsio Download Center, http://service.chelsio.com/

ii. Untar the tarball using the following command:

```
[root@host~]# tar zxvfm <driver_package>.tar.gz
```

iii. Change your current working directory to Chelsio Unified Wire package directory and build the source using:

```
[root@host~]# make
```

iv. Install the drivers, tools and libraries using the following command:

```
[root@host~]# make install
```

v. The default configuration tuning option is *Unified Wire.* The configuration tuning can be selected using the following commands:

```
[root@host~]# make CONF=<configuration_tuning>
[root@host~]# make CONF=<configuration_tuning> install
```

Important *Steps (iv) and (v) mentioned above will NOT install DCB driver and benchmark tools. They will have to be installed manually.*

Please refer to section **CLI mode (individual drivers)** *for instructions on installing them.*

Note *To view the different configuration tuning options, view help by typing* `[root@host~]# make help`

vi. Reboot your machine for changes to take effect.

3.4.4.1. Installation on updated kernels

If the kernel version on your Linux distribution is updated, please execute the following command to install the Unified Wire package:

```
[root@host~]# make UNAME_R=<kernel_version>
```

Where `kernel_version` is the nearest supported kernel version.

For example, if you want to install the package on a RHEL 6 distribution updated to 2.6.32-431.20.3. el6 kernel, run the following commands:

```
[root@host~]# make UNAME_R=2.6.32-431.el6
[root@host~]# make UNAME_R=2.6.32-431.el6 install
```

To view the list of the supported kernel versions, run the following command:

```
[root@host~]# make list_kernels
```

Reboot your machine for changes to take effect.

3.4.5. CLI mode (individual drivers)

You can also choose to install drivers individually. Provided here are steps to build and install NIC, TOE, iWARP, WD-UDP, UDP Segmentation Offload, Crypto Offload, DCB drivers and benchmarking tools. To know about other drivers, view help by running `make help`.

- To build and install NIC driver without offload support:

```
[root@host~]# make nic
[root@host~]# make nic_install
```

- To build and install NIC driver with offload support and Offload drivers:

```
[root@host~]# make toe
[root@host~]# make toe_install
```

- To build and install Offload drivers without IPv6 support:

```
[root@host~]# make toe_ipv4
[root@host~]# make toe_ipv4_install
```

- To build and install iWARP driver against outbox OFED:

```
[root@host~]# make iwarp
[root@host~]# make iwarp_install
```

- To build and install all drivers without IPv6 support:

```
[root@host~]# make ipv6_disable=1
[root@host~]# make ipv6_disable=1 install
```

- To build and install Crypto Offload driver

```
[root@host~]# make crypto
[root@host~]# make crypto_install
```

- To build and install all drivers with DCB support:

```
[root@host~]# make dcbx=1
[root@host~]# make dcbx=1 install
```

- The offload drivers support UDP Segmentation Offload with limited number of connections (1024 connections). To build and install UDP Offload drivers which support large number of offload connections (approx 10K):

```
[root@host~]# make udp_offload
[root@host~]# make udp_offload_install
```

- The default configuration tuning option is *Unified Wire.* The configuration tuning can be selected using the following commands:

```
[root@host~]# make CONF=<configuration_tuning> <Build Target>
[root@host~]# make CONF=<configuration_tuning> <Install Target>
```

- To build and install drivers along with benchmarks:

```
[root@host~]# make BENCHMARKS=1
[root@host~]# make BENCHMARKS=1 install
```

- Unified Wire Manager will be installed by default. To skip the installation:

```
[root@host~]# make INSTALL_UM=0 install
```

- The drivers will be installed as RPMs or Debian packages (for ubuntu). To skip this and install drivers:

```
[root@host~]# make SKIP_RPM=1 install
```

Note *To view the different configuration tuning options, view the help by typing* `[root@host~]#make help`

Note *If IPv6 is administratively disabled in the machine, the drivers will be built and installed without IPv6 Offload support by default.*

3.5. Installing Chelsio Unified Wire from RPM

Note *Drivers installed from RPM Packages do not have DCB support.*

3.5.1. Menu Mode

i. Download the tarball specific to your operating system and architecture from Chelsio Download Center, http://service.chelsio.com/

ii. Untar the tarball:
E.g. for RHEL 6.6, untar using the following command:

```
[root@host~]# tar zxvfm <driver_package>-RHEL6.6_x86_64.tar.gz
```

iii. Change your current working directory to Chelsio Unified Wire package directory and run the following command:

```
[root@host~]# ./install.py
```

iv. Select the Installation type as described below. Enter the corresponding number in the Input field and press Enter.
   1. *Unified Wire*: Install all the drivers in the Unified Wire software package. This option will not install OFED and drivers built against OFED.
   2. *T5 Wire Direct Latency*: Install Wire Direct Latency drivers needed for Low latency applications.
   3. *Custom*: Customize the installation. Use this option to install drivers/software and related components (like OFED-3.18-2) according to the tuning option selected.
   4. *EXIT*: Exit the installer.

Note *The Installation options may vary depending on the Configuration tuning option selected.*

v. The selected components will now be installed.
vi. Reboot your machine for changes to take effect.

Note *If the installation aborts with the message "Resolve the errors/dependencies manually and restart the installation", please go through the install.log to resolve errors/dependencies and then start the installation again.*

3.5.2. CLI mode

i. Download the tarball specific to your operating system and architecture from Chelsio Download Center, http://service.chelsio.com/

ii. Untar the tarball:

E.g. For RHEL 6.6, untar using the following command:

```
[root@host~]# tar zxvfm <driver_package>-RHEL6.6_x86_64.tar.gz
```

iii. Change your current working directory to Chelsio Unified Wire package directory and install Unified Wire using:

```
[root@host~]# ./install.py -i <nic_toe/all/udpso/wd>
```

nic_toe : NIC and TOE drivers only
all : all Chelsio drivers built against inbox OFED
udpso : UDP segmentation offload capable NIC and TOE drivers only
wd : Wire Direct drivers and libraries only

Note *The Installation options may vary depending on the Linux distribution.*

iv. The default configuration tuning option is *Unified Wire*. The configuration tuning can be selected using the following command:

```
[root@host~]# ./install.py -i <Installation mode> -c <configuration_tuning>
```

Note *To view the different configuration tuning options, view the help by typing*
`[root@host~]# ./install.py -h`

v. To install OFED and Chelsio drivers built against OFED, run the above command with *-o* option.

```
[root@host~]# ./install.py -i <Installation mode> -c <Configuration> -o
```

vi. Reboot your machine for changes to take effect.

3.5.2.1. iWARP driver installation on cluster nodes

Important *Please make sure that you have enabled password less authentication with ssh on the peer nodes for this feature to work.*

i. Create a file (*machinefilename*) containing the IP addresses or hostnames of the nodes in the cluster. You can view the sample file, *sample_machinefile,* provided in the package to view the format in which the nodes have to be listed.

ii. Navigate to driver package directory and execute the following command:

```
[root@host~]# ./install.py -C  -m <machinefilename> -i
<nic_toe/all/udpso/wd> -c <configuration_tuning> -o
```

Here, -o parameter will install OFED and Chelsio drivers built against OFED.

The above command will install iWARP (*iw_cxgb4*) and TOE (*t4_tom*) drivers on all the nodes listed in the <machinefilename> file.

iii. Reboot your machine for changes to take effect.

3.6. Firmware update

The firmware is installed on the system, typically in `/lib/firmware/cxgb4`, and the driver will auto-load the firmware if an update is required. The kernel must be configured to enable userspace firmware loading support:

Device Drivers -> Generic Driver Options -> Userspace firmware loading support

The firmware version can be verified using *ethtool*:

```
[root@host~]# ethtool -i <iface>
```

4. Software/Driver Uninstallation

Similar to installation, the Chelsio Unified Wire package can be uninstalled using two main methods: from the source and RPM, based on the method used for installation. If you decide to use source, you can uninstall the package using CLI or GUI mode.

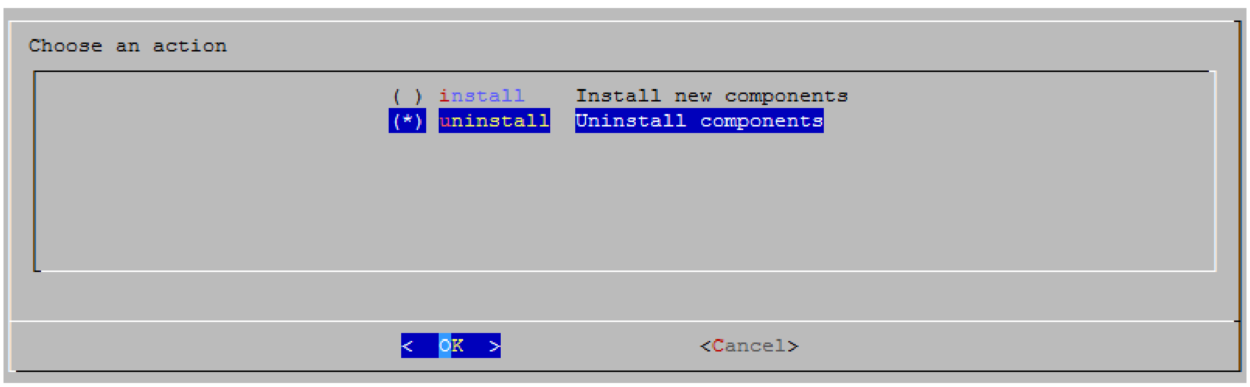
4.1. Uninstalling Chelsio Unified Wire from source

4.1.1. GUI mode (with Dialog utility)

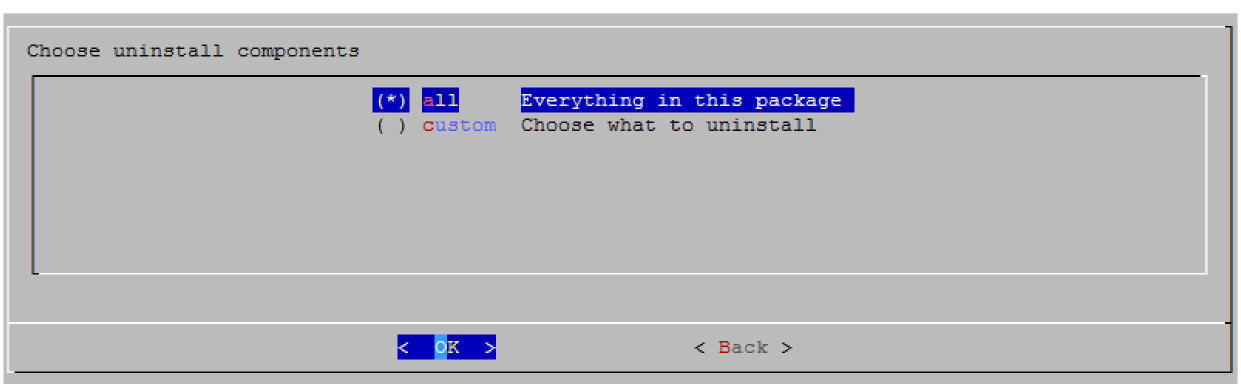
i. Change your current working directory to Chelsio Unified Wire package directory and run the following script to start the GUI installer:

```
[root@host~]# ./install.py
```

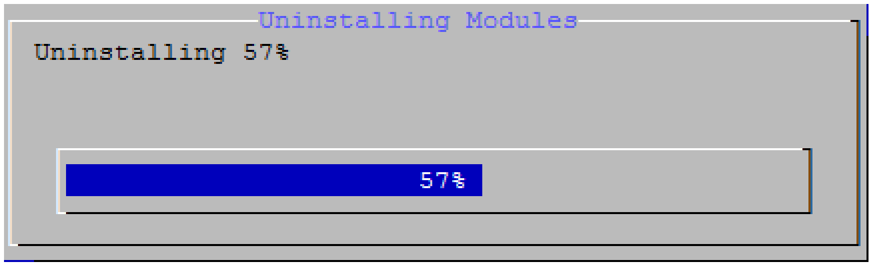
ii. Select “uninstall” , Under “Choose an action”



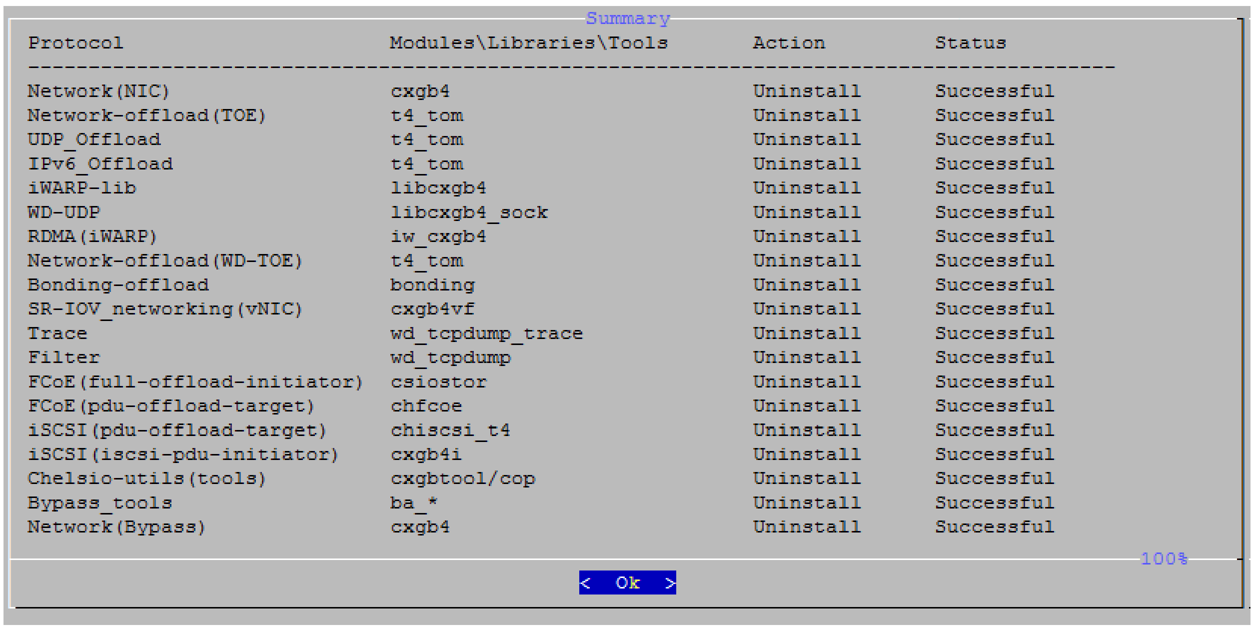
iii. Select “all” to uninstall all the installed drivers, libraries and tools or select “custom” to remove specific components.



iv. The selected components will now be uninstalled.



v. After successful uninstalltion, summary of the uninstalled components will be displayed.

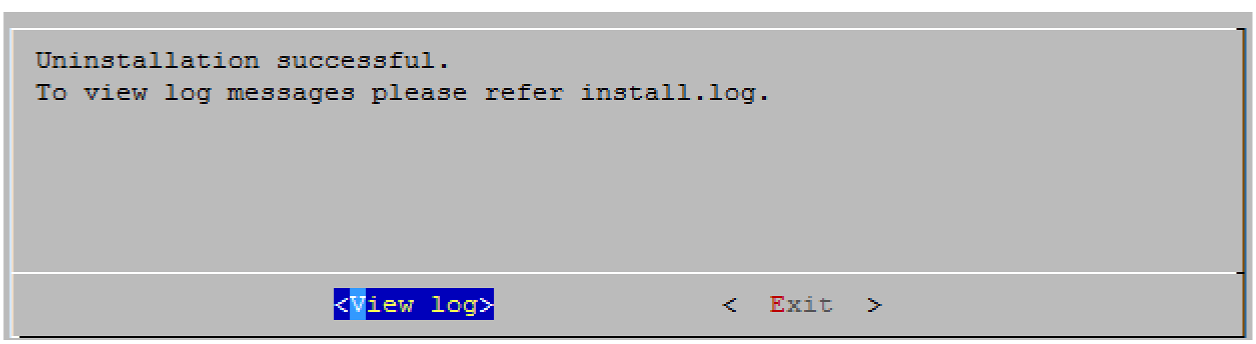

Summary

Protocol	Modules\Libraries\Tools	Action	Status
Network(NIC)	cxgb4	Uninstall	Successful
Network-offload(TOE)	t4_tom	Uninstall	Successful
UDP_Offload	t4_tom	Uninstall	Successful
IPv6_Offload	t4_tom	Uninstall	Successful
iWARP-lib	libcxgb4	Uninstall	Successful
WD-UDP	libcxgb4_sock	Uninstall	Successful
RDMA(iWARP)	iw_cxgb4	Uninstall	Successful
Network-offload(WD-TOE)	t4_tom	Uninstall	Successful
Bonding-offload	bonding	Uninstall	Successful
SR-IOV_networking(vNIC)	cxgb4vf	Uninstall	Successful
Trace	wd_tcpdump_trace	Uninstall	Successful
Filter	wd_tcpdump	Uninstall	Successful
FCoE(full-offload-initiator)	csiostor	Uninstall	Successful
FCoE(pdu-offload-target)	chfcoe	Uninstall	Successful
iSCSI(pdu-offload-target)	chiscsi_t4	Uninstall	Successful
iSCSI(iscsi-pdu-initiator)	cxgb4i	Uninstall	Successful
Chelsio-utils(tools)	cxgbtool/cop	Uninstall	Successful
Bypass_tools	ba_*	Uninstall	Successful
Network(Bypass)	cxgb4	Uninstall	Successful

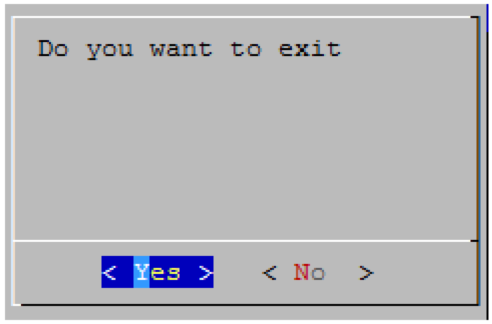
100%

< Ok >

vi. Select “View log” to view uninstallation log or “Exit” to continue.



vii. Select “Yes” to exit the installer or “No” to go back.



Note *Press Esc or Ctrl+C to exit the installer at any point of time.*

4.1.2. CLI mode (without Dialog utility)

Run the following script with `-u` option to uninstall the Unified Wire Package:

```
[root@host~]# ./install.py -u <target>
```

Note *View help by typing* `[root@host~]# ./install.py -h` for more information

4.1.3. CLI mode

Change your current working directory to Chelsio Unified Wire package directory and uninstall using the following command:

```
[root@host~]# make uninstall
```

Note *Uninstalling Unified Wire package will not uninstall Unified Wire Manager. Refer to the next section,* **CLI mode (individual drivers)** *to remove the software manually.*

4.1.4.1. iWARP driver uninstallation on Cluster nodes

To uninstal iWARP drivers on multiple Cluster nodes with a single command, run the following command:

```
[root@host~]# ./install.py -C -m <machinefilename> -u all
```

The above command will remove Chelsio iWARP (*iw_cxgb4*) and TOE (*t4_tom*) drivers from all the nodes listed in the *machinefilename* file.

4.1.5. CLI mode (individual drivers/software)

You can also choose to uninstall drivers/software individually. Provided here are steps to uninstall NIC, TOE, iWARP, UDP Segmentation Offload drivers and Unified Wire Manager (UM). To know about other drivers, access help by running `make help`

- To uninstall NIC driver:

```
[root@host~]# make nic_uninstall
```

- To uninstall offload driver:

```
[root@host~]# make toe_uninstall
```

- To uninstall iWARP driver:

```
[root@host~]# make iwarp_uninstall
```

- To uninstall UDP Segmentation Offload driver:

```
[root@host~]# make udp_offload_uninstall
```

- To uninstall Unified Wire Manager (UM):

```
[root@host~]# make uninstall UM_UNINST=1
```

OR

```
[root@host~]# make tools_uninstall UM_UNINST=1
```

4.2. Uninstalling Chelsio Unified Wire from RPM

Change your current working directory to Chelsio Unified Wire package directory and and run the following command:

```
[root@host~]# ./uninstall.py <inbox/ofed>
```

inbox : for removing all Chelsio drivers.
ofed : for removing OFED and Chelsio drivers.

Note *The uninstallation options may vary depending on Linux distribution. View help by typing* `[root@host~]# ./uninstall.py -h` *for more information.*

Note *Uninstalling Unified Wire package will not uninstall Unified Wire Manager. Refer to the* ***Unified Wire Manager (UM)*** *chapter to remove the software manually (Click here).*

4.2.1.1. iWARP driver uninstallation on Cluster nodes

To uninstal iWARP drivers on multiple Cluster nodes with a single command, run the following:

```
[root@host~]# ./install.py -C -m <machinefilename> -u
```

The above command will remove Chelsio iWARP (*iw_cxgb4*) and TOE (*t4_tom*) drivers from all the nodes listed in the *machinefilename* file.

5. Configuring Chelsio Network Interfaces

In order to test Chelsio adapters' features it is required to use two machines both with Chelsio's network adapters installed. These two machines can be connected directly without a switch (back-to-back), or both connected to a switch. The interfaces have to be declared and configured. The configuration files for network interfaces on Red Hat Enterprise Linux (RHEL) distributions are kept under `/etc/sysconfig/network-scripts`.

Note *Some operating systems may attempt to auto-configure the detected hardware and some may not detect all ports on a multi-port adapter. If this happens, please refer to the operating system documentation for manually configuring the network device.*

5.1. Configuring 40G adapters

Chelsio T5 40G adapters can be configured in the following three modes:

i. 2X40Gbps: This is the default mode of operation where each port functions as 40Gbps link. The port nearest to the motherboard will appear as the first network interface (Port 0).
ii. 4X10Gbps: In this mode, port 0 functions as 4 10Gbps links and port 1 is disabled.
iii. QSA: This mode adds support for QSA (QSFP to SFP+) modules, enabling smooth, cost-effective, connections between 40 Gigabit Ethernet adapters and 1 or 10 Gigabit Ethernet networks using existing SFP+ based cabling. The port farthest from the motherboard will appear as the first network interface (Port 0).

To configure/change the mode of operation, use the following procedure:

i. Unload all Chelsio drivers using the *rmmod* command:

```
[root@host~]# rmmod <chelsio_driver>
```

ii. Run the *chelsio_adapter_config* command to detect all T5 40G adapter(s) present in the system.

```
[root@host~]# chelsio_adapter_config
Chelsio T580 card detected
Chelsio T580 PCI devices :
|------------------------------|
| 1 T580-LP-CR     01:00.0      |
| 2 T580-CR        03:00.0      |
| 3 T580-LP-SO-CR 04:00.0      |
|------------------------------|
```

iii. Select the adapter to configure by specifying the adapter index.

iv. Select the required mode:

```
Possible T580 adapter modes:
|------------------------------|
| 1: 2x40G                     |
| 2: 4x10G                     |
| 3: QSA                       |
|------------------------------|
Select mode for adapter (1,2,3):
```

v. Reload the network driver for changes to take effect.

```
[root@host~]# rmmod cxgb4
[root@host~]# modprobe cxgb4
```

Note *In case of T580-SO-CR adapters, reboot the machine for changes to take effect.*

5.2. Configuring network-scripts

A typical interface network-script (e.g. eth0) on RHEL 6.X looks like the following:

```
# file: /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE="eth0"
HWADDR=00:30:48:32:6A:AA
ONBOOT="yes"
NM_CONTROLLED="no"
BOOTPROTO="static"
IPADDR=10.192.167.111
NETMASK=255.255.240.0
```

Note *On earlier versions of RHEL the NETMASK attribute is named IPMASK. Make sure you are using the right attribute name.*

In the case of DHCP addressing the last two lines should be removed and `BOOTPROTO="static"` should be changed to `BOOTPROTO="dhcp"`

The `ifcfg-ethX` files have to be created manually. They are required for bringing the interfaces up and down and attribute the desired IP addresses.

5.3. Creating network-scripts

To spot the new interfaces, make sure the driver is unloaded first. To that point `ifconfig -a | grep HWaddr` should display all non-chelsio interfaces whose drivers are loaded, whether the interfaces are up or not.

```
[root@host~]# ifconfig -a | grep HWaddr
eth0 Link encap:Ethernet HWaddr 00:30:48:32:6A:AA
```

Then load the driver using the `modprobe cxgb4` command (for the moment it does not make any difference whether we are using NIC-only or the TOE-enabling driver). The output of ifconfig should display the adapter interfaces as:

```
[root@host~]# ifconfig -a | grep HWaddr
eth0 Link encap:Ethernet HWaddr 00:30:48:32:6A:AA
eth1 Link encap:Ethernet HWaddr 00:07:43:04:6B:E9
eth2 Link encap:Ethernet HWaddr 00:07:43:04:6B:F1
```

For each interface you can write a configuration file in `/etc/sysconfig/network-scripts`. The `ifcfg-eth1` could look like:

```
# file: /etc/sysconfig/network-scripts/ifcfg-eth1
DEVICE="eth1"
HWADDR=00:07:43:04:6B:E9
ONBOOT="no"
NM_CONTROLLED="no"
BOOTPROTO="static"
IPADDR=10.192.167.112
NETMASK=255.255.240.0
```

From now on, the `eth1` interface of the adapter can be brought up and down through the `ifup eth1` and `ifdown eth1` commands respectively. Note that it is of course not compulsory to create a configuration file for every interface if you are not planning to use them all.

5.4. Checking Link

Once the network-scripts are created for the interfaces you should check the link i.e. make sure it is actually connected to the network. First, bring up the interface you want to test using `ifup eth1`.

You should now be able to ping any other machine from your network provided it has ping response enabled.

6. Performance Tuning

In order to auto tune the system for best performance, Chelsio recommends:

- Disabling virtualization, c-state technology, VT-d, Intel I/O AT and SR-IOV in the BIOS settings
- Installing the adapter into a PCIe Gen3 x8/x16 slot.
- Installing the **tools** which will copy `t4_perftune.sh` script to **/sbin** directory. Run the script to map the adapter queues to different CPUs:

```
[root@host~]# t4_perftune.sh
```

Also, follow the steps mentioned below to lower your latency:

i. Disable SELinux
ii. Run the following script to disable few services.

```
[root@host~]# t4_latencytune.sh <interface>
```

iii. Set sysctl param *net.ipv4.tcp_low_latency* to 1

```
[root@host~]# sysctl -w net.ipv4.tcp_low_latency=1
```

To optimize your system for different protocols, please refer to their respective chapters.

7. Software/Driver Update

For any distribution specific problems, please check README and Release Notes included in the release for possible workaround.

Please visit Chelsio support web site http://service.chelsio.com/ for regular updates on various software/drivers. You can also subscribe to our newsletter for the latest software updates.

II. Network (NIC/TOE)

1. Introduction

Chelsio's Unified Wire adapters provide extensive support for NIC operation, including all stateless offload mechanisms for both IPv4 and IPv6 (IP, TCP and UDP checksum offload, LSO - Large Send Offload aka TSO - TCP Segmentation Offload, and assist mechanisms for accelerating LRO - Large Receive Offload).

A high performance fully offloaded and fully featured TCP/IP stack meets or exceeds software implementations in RFC compliance. Chelsio's Terminator engine provides unparalleled performance through a specialized data flow processor implementation and a host of features designed for high throughput and low latency in demanding conditions and networking environments.

TCP offload is fully implemented in the hardware, thus freeing the CPU from TCP/IP overhead. The freed CPU can be used for any computing needs. The TCP offload in turn removes network bottlenecks and enables applications to take full advantage of the networking capabilities.

1.1. Hardware Requirements

1.1.1. Supported adapters

The following are the currently shipping Chelsio adapters that are compatible with Chelsio Network driver:

- T62100-LP-CR
- T6225-CR
- T580-OCP-SO*
- T520-OCP-SO*
- T520-BT
- T580-CR
- T580-SO-CR*
- T580-LP-CR
- T520-LL-CR
- T520-SO-CR*
- T520-CR
- T540-CR
- T420-CR
- T440-CR
- T422-CR
- T420-SO-CR*
- T404-BT
- T420-BCH
- T440-LP-CR
- T420-BT

- T420-LL-CR
- T420-CX

*Only NIC driver supported

1.2. Software Requirements

1.2.1. Linux Requirements

Currently the Network driver is available for the following versions:

- RHEL 7.3, 3.10.0-514.el7
- RHEL 7.3, 3.10.0-514.el7.ppc64le (POWER8 LE)
- RHEL 7.3, 4.5.0-15.el7.aarch64 (ARM64)
- RHEL 7.2, 3.10.0-327.el7
- RHEL 6.8, 2.6.32-642.el6
- RHEL 6.7, 2.6.32-573.el6
- SLES 12 SP2, 4.4.21-69-default
- SLES 12 SP1, 3.12.49-11-default
- SLES 11 SP4, 3.0.101-63-default
- Ubuntu 16.04.1, 4.4.0-47-generic
- Ubuntu 14.04.4, 4.2.0-27-generic
- Kernel.org linux-4.8

Other kernel versions have not been tested and are not guaranteed to work.

2. Software/Driver Loading

Important *Please ensure that all inbox drivers are unloaded before proceeding with unified wire drivers.*

The driver must be loaded by the root user. Any attempt to load the driver as a regular user will fail.

2.1. Loading in NIC mode (without full offload support)

To load the Network driver without full offload support, run the following command:

```
[root@host~]# modprobe cxgb4
```

2.2. Loading in TOE mode (with full offload support)

To enable full offload support, run the following command:

```
[root@host~]# modprobe t4_tom
```

Note *Offload support needs to be enabled upon each reboot of the system. This can be done manually as shown above.*

In VMDirect Path environment, it is recommended to load the offload driver using the following command:

```
[root@host~]# modprobe t4_tom vmdirectio=1
```

3. Software/Driver Unloading

3.1. Unloading the NIC driver

To unload the NIC driver, run the following command:

```
[root@host~]# rmmod cxgb4
```

3.2. Unloading the TOE driver

A reboot is required to unload the TOE driver. To avoid rebooting, follow the steps mentioned below:

i. Load *t4_tom* driver with *unsupported_allow_unload* parameter.

```
[root@host~]# modprobe t4_tom unsupported_allow_unload=1
```

ii. Stop all the offloaded traffic, servers and connections. Check for the reference count.

```
[root@host~]# cat /sys/module/t4_tom/refcnt
```

If the reference count is 0, the driver can be directly unloaded. Skip to step (iii)

If the count is non-zero, load a COP policy which disables offload using the following procedure:

a. Create a policy file which will disable offload

```
[root@host~]# cat policy_file
all => !offload
```

b. Compile and apply the output policy file

```
[root@host~]# cop -o no-offload.cop policy_file
[root@host~]# cxgbtool ethX policy no-offload.cop
```

iii. Unload the driver:

```
[root@host~]# rmmod t4_tom
[root@host~]# rmmod toecore
[root@host~]# rmmod cxgb4
```

4. Software/Driver Configuration and Fine-tuning

4.1. Instantiate Virtual Functions (SR-IOV)

To instantiate the Virtual functions, load the cxgb4 driver with `num_vf` parameter with a non-zero value. For example:

```
[root@host~]# modprobe cxgb4 num_vf=1,0,0,0
```

The number(s) provided for `num_vf` parameter specifies the number of Virtual Functions to be instantiated per Physical Function. The Virtual Functions can be assigned to Virtual Machines (Guests). A maximum of 64 Virtual Functions can be instantiated with 16 Virtual Functions per Physical Function. Loading the *cxgb4* driver with `num_vf` parameter loads the *cxgb4vf* module (the driver for Virtual Functions) in the host by default. Hence unload the *cxgb4vf* module (on the host) before assigning Virtual Functions to the Virtual Machines (Guests), using the following command:

```
[root@host~]# rmmod cxgb4vf
```

Note *To get familiar with physical and virtual function terminologies, please refer the PCI Express specification.*

4.2. Enabling Busy waiting

Busy waiting/polling is a technique where a process repeatedly checks to see if an event has occurred, by spinning in a tight loop. By making use of similar technique, Linux kernel provides the ability for the socket layer code to poll directly on an Ethernet device's Rx queue. This eliminates the cost of interrupts and context switching, and with proper tuning allows to achieve latency performance similar to that of hardware.

Chelsio's NIC and TOE drivers support this feature and can be enabled on Chelsio supported devices to attain improved latency.

To make use of BUSY_POLL feature, follow the steps mentioned below:

i. Enable BUSY_POLL support in kernel config file by setting `CONFIG_NET_RX_BUSY_POLL=y`
ii. Enable BUSY_POLL globally in the system by setting the values of following sysctl parameters depending on the number of connections:

```
sysctl -w net.core.busy_read=<value>
sysctl -w net.core.busy_poll=<value>
```

Set the values of the above parameters to 50 for 100 or less connections; and 100 for more than 100 connections.

Note *BUSY_POLL can also be enabled on a per-connection basis by making use of SO_BUSY_POLL option in the socket application code. Refer socket man-page for more details.*

4.3. Performance Tuning

- **Receiver Side Scaling (RSS)**

Receiver Side Scaling enables the receiving network traffic to scale with the available number of processors on a modern networked computer. RSS enables parallel receive processing and dynamically balances the load among multiple processors. Chelsio's network controller fully supports Receiver Side Scaling for IPv4 and IPv6.

This script first determines the number of CPUs on the system and then each receiving queue is bound to an entry in the system interrupt table and assigned to a specific CPU. Thus, each receiving queue interrupts a specific CPU through a specific interrupt now. For example, on a 4-core system, `t4_perftune.sh` gives the following output:

```
[root@host~]# t4_perftune.sh
Discovering Chelsio T4/T5 devices ...
Configuring Chelsio T4/T5 devices ...
Tuning eth7
IRQ table length 4
Writing 1 in /proc/irq/62/smp_affinity
Writing 2 in /proc/irq/63/smp_affinity
Writing 4 in /proc/irq/64/smp_affinity
Writing 8 in /proc/irq/65/smp_affinity
eth7 now up and tuned
...
```

Because there are 4 CPUs on the system, 4 entries of interrupts are assigned. For other network interfaces, you should see similar output message.

Now the receiving traffic is dynamically assigned to one of the system's CPUs through a Terminator queue. This achieves a balanced usage among all the processors. This can be verified, for example, by using the **iperf** tool. First set up a server on the receiver host:

```
[root@receiver_host~]# iperf -s
```

Then on the sender host, send data to the server using the iperf client mode. To emulate a moderate traffic workload, use *-P* option to request 20 TCP streams from the server:

```
[root@sender_host~]# iperf -c receiver_host_name_or_IP -P 20
```

Then on the receiver host, look at interrupt rate at `/proc/interrupts`:

```
[root@receiver_host~]# cat /proc/interrupts | grep eth6

Id       CPU0       CPU1       CPU2       CPU3       type             interface
36:    115229          0          0          1   PCI-MSI-edge     eth6 (queue 0)
37:         0     121083          1          0   PCI-MSI-edge     eth6 (queue 1)
38:         0          0     105423          1   PCI-MSI-edge     eth6 (queue 2)
39:         0          0          0     115724   PCI-MSI-edge     eth6 (queue 3)
```

Now interrupts from eth6 are evenly distributed among the 4 CPUs.

Without Terminator's RSS support, the interrupts caused by network traffic may be distributed unevenly over CPUs. For your information, the traffic produced by the same iperf commands gives the following output in `/proc/interrupts`.

```
[root@receiver_host~]# cat /proc/interrupts | grep eth6

Id       CPU0       CPU1       CPU2       CPU3       type             interface
36:         0          9          0      17418  PCI-MSI-edge    eth6 (queue 0)
37:         0          0      21718       2063  PCI-MSI-edge    eth6 (queue 1)
38:         0          7     391519        222  PCI-MSI-edge    eth6 (queue 2)
39:         1          0         33      17798  PCI-MSI-edge    eth6 (queue 3)
```

Here there are 4 receiving queues from the eth6 interface, but they are not bound to a specific CPU or interrupt entry. Queue 2 has caused a very large number of interrupts on CPU2 while CPU0 and CPU1 are barely used by any of the four queues. Enabling RSS is thus essential for best performance.

Note *Linux's* `irqbalance` *may take charge of distributing interrupts among CPUs on a multiprocessor platform. However,* `irqbalance` *distributes interrupt requests from all hardware devices across processors. For a server with Chelsio network card constantly receiving large volume of data at 40/10Gbps, the network interrupt demands are significantly high. Under such circumstances, it is necessary to enable RSS to balance the network load across multiple processors and achieve the best performance.*

- **Interrupt Coalescing**

The idea behind Interrupt Coalescing (IC) is to avoid flooding the host CPUs with too many interrupts. Instead of throwing one interrupt per incoming packet, IC waits for 'n' packets to be available in the Rx queues and placed into the host memory through DMA operations before an interrupt is thrown, reducing the CPU load and thus improving latency. It can be changed using the following command:

```
[root@host~]# ethtool –C ethX rx-frames n
```

Note *For more information, run the following command:*
`[root@host~]# ethtool -h`

- **Configuring sysctl, adaptive interrupts, select_queue (NIC)**

i. Turn off *irqbalance*

```
[root@host~]# /etc/init.d/irqbalance stop
```

ii. Add the following sysctl parameters to */etc/sysctl.conf*

```
sysctl -w net.ipv4.tcp_timestamps=0
sysctl -w net.ipv4.tcp_low_latency=1
sysctl -w net.core.netdev_max_backlog=250000
sysctl -w net.core.rmem_max=16777216
sysctl -w net.core.wmem_max=16777216
sysctl -w net.core.rmem_default=16777216
sysctl -w net.core.wmem_default=16777216
sysctl -w net.core.optmem_max=16777216
sysctl -w net.ipv4.tcp_rmem='4096 87380 16777216'
sysctl -w net.ipv4.tcp_wmem='4096 65536 16777216'
```

iii. Bring up the network interfaces and run the following command:

```
[root@host~]# ethtool -C ethXX adaptive-rx on
```

Read back the *ethtool* settings with the following command:

```
[root@host~]# ethtool -c ethXX
```

Output should show *adaptive-rx* as on.

iv. Change *select_queue* parameter's value to 1:

```
[root@host~]# cat /sys/module/cxgb4/parameters/select_queue
0
[root@host~]# echo 1 > /sys/module/cxgb4/parameters/select_queue
[root@host~]# cat /sys/module/cxgb4/parameters/select_queue
1
```

For **TOE** performance, follow the first two steps mentioned above and then set the following *sysctl* parameter:

```
[root@host~]# sysctl -w toe.toe0_tom.delayed_ack=3
```

- **Large Receive Offload / Generic Receive Offload**

Large Receive Offload or Generic Receive Offload is a performance improvement feature at the receiving side. LRO/GRO aggregates the received packets that belong to same stream, and combines them to form a larger packet before pushing them to the receive host network stack. By doing this, rather than processing every small packet, the receiver CPU works on fewer packet headers but with same amount of data. This helps reduce the receive host CPU load and improve throughput in a 40/10Gb network environment where CPU can be the bottleneck.

LRO and GRO are different names to refer to the same receiver packets aggregating feature. LRO and GRO actually differ in their implementation of the feature in the Linux kernel. The feature was first added into the Linux kernel in version 2.6.24 and named Large Receive Offload (LRO). However, LRO only works for TCP and IPv4. As from kernel 2.6.29, a new protocol-independent implementation removing the limitation is added to Linux, and it is named Generic Receive Offload (GRO). The old LRO code is still available in the kernel sources but whenever both GRO and LRO are presented GRO is always the preferred one to use.

Please note that if your Linux system has IP forwarding enabled, i.e. acting as a bridge or router, the LRO needs to be disabled. This is due to a known kernel issue.

Chelsio's card supports both hardware assisted GRO/LRO and Linux-based GRO/LRO. `t4_tom` is the kernel module that enables the hardware assisted GRO/LRO. If it is not already in the kernel module list, use the following command to insert it:

```
[root@host~]# lsmod | grep t4_tom
[root@host~]# modprobe t4_tom

[root@host~]# lsmod | grep t4_tom
t4_tom 88378 0 [permanent]
toecore 21618 1 t4_tom
cxgb4 225342 1 t4_tom
```

Then Terminator's hardware GRO/LRO implementation is enabled.

If you would like to use the Linux GRO/LRO for any reason, first the `t4_tom` kernel module needs to be removed from kernel module list. Please note you might need to reboot your system.

After removing the `t4_tom` module, you can use `ethtool` to check the status of current GRO/LRO settings, for example:

```
[root@host~]# ethtool -k eth6
Offload parameters for eth6:
rx-checksumming: on
tx-checksumming: on
scatter-gather: on
tcp-segmentation-offload: on
udp-fragmentation-offload: off
generic-segmentation-offload: on
generic-receive-offload: on
large-receive-offload: off
```

Now the `generic-receive-offload` option is on. This means GRO is enabled. Please note that there are two offload options here: `generic-receive-offload` and `large-receive-offload`. This is because on this Linux system (RHEL6.0), the kernel supports both GRO and LRO. As mentioned earlier, GRO is always the preferred option when both of them are present. On other systems LRO might be the only available option. Then `ethtool` could be used to switch LRO on and off as well.

When Linux's GRO is enabled, Chelsio's driver provides two GRO-related statistics. They are displayed using the following command:

```
[root@host~]# ethtool -S eth6
...
GROPackets : 0
GROMerged : 897723
...
```

`GROPackets` is the number of held packets. Those are candidate packets held by the kernel to be processed individually or to be merged to larger packets. This number is usually zero. `GROMerged` is the number of packets that merged to larger packets. Usually this number increases if there is any continuous traffic stream present. `ethtool` can also be used to switch off the GRO/LRO options when necessary:

```
[root@host~]# ethtool -K eth6 gro off
[root@host~]# ethtool -k eth6
Offload parameters for eth6:
rx-checksumming: on
tx-checksumming: on
scatter-gather: on
tcp-segmentation-offload: on
udp-fragmentation-offload: off
generic-segmentation-offload: on
generic-receive-offload: off
large-receive-offload: off
```

The output above shows a disabled GRO.

III. Virtual Function Network (vNIC)

1. Introduction

The ever increasing network infrastructure of IT enterprises has lead to a phenomenal increase in maintenance and operational costs. IT managers are forced to acquire more physical servers and other data center resources to satisfy storage and network demands. To solve the Network and I/O overhead, users are opting for server virtualization which consolidates I/O workloads onto lesser physical servers thus resulting in efficient, dynamic and economical data center environments. Other benefits of Virtualization include improved disaster recovery, server portability, cloud computing, Virtual Desktop Infrastructure (VDI), etc.

Chelsio's Unified Wire family of adapters deliver increased bandwidth, lower latency and lower power with virtualization features to maximize cloud scaling and utilization. The adapters also provide full support for PCI-SIG SR-IOV to improve I/O performance on a virtualized system. User can configure up to 64 Virtual and 8 Physical functions (with 4 PFs as SR-IOV capable) along with 336 virtual MAC addresses.

1.1. Hardware Requirements

1.1.1. Supported adapters

The following are the currently shipping Chelsio adapters that are compatible with the Chelsio vNIC driver:

- T62100-LP-CR
- T6225-CR
- T520-BT
- T580-CR
- T520-LL-CR
- T520-CR
- T580-LP-CR
- T540-CR
- T420-CR
- T440-CR
- T422-CR
- T420-SO-CR
- T404-BT
- T440-LP-CR
- T420-BT
- T420-LL-CR
- T420-CX

1.2. Software Requirements

1.2.1. Linux Requirements

Currently the vNIC driver is available for the following versions:

- RHEL 7.3, 3.10.0-514.el7
- RHEL 7.2, 3.10.0-327.el7
- RHEL 6.8, 2.6.32-642.el6
- RHEL 6.7, 2.6.32-573.el6
- SLES 12 SP2, 4.4.21-69-default
- SLES 12 SP1, 3.12.49-11-default
- SLES 11 SP4, 3.0.101-63-default
- Ubuntu 16.04.1, 4.4.0-47-generic
- Ubuntu 14.04.4, 4.2.0-27-generic
- Kernel.org linux-4.8

Other kernel versions have not been tested and are not guaranteed to work.

2. Software/Driver Loading

Important *Please ensure that all inbox drivers are unloaded before proceeding with unified wire drivers.*

2.1. Instantiate Virtual Functions

To instantiate Chelsio Virtual Functions, please refer the Network (NIC/TOE) section.

2.2. Loading the driver

The vNIC driver must be loaded on the Guest OS by the root user. Any attempt to load the driver as a regular user will fail.

To load the driver, run the following command:

```
[root@host~]# modprobe cxgb4vf
```

3. Software/Driver Unloading

3.1. Unloading the driver

The vNIC driver must be unloaded on the Guest OS by the root user. Any attempt to unload the driver as a regular user will fail.

To unload the driver, execute the following command:

```
[root@host~]# rmmod cxgb4vf
```

IV. iWARP (RDMA)

1. Introduction

Chelsio's Terminator engine implements a feature rich RDMA implementation which adheres to the IETF standards with optional markers and MPA CRC-32C.

The iWARP RDMA operation benefits from the virtualization, traffic management and QoS mechanisms provided by Terminator engine. It is possible to ACL process iWARP RDMA packets. It is also possible to rate control the iWARP traffic on a per-connection or per-class basis, and to give higher priority to QPs that implement distributed locking mechanisms. The iWARP operation also benefits from the high performance and low latency TCP implementation in the offload engine.

1.1. Hardware Requirements

1.1.1. Supported adapters

The following are the currently shipping Chelsio adapters that are compatible with Chelsio iWARP driver:

- T62100-LP-CR
- T6225-CR
- T520-BT
- T580-CR
- T520-LL-CR
- T520-CR
- T580-LP-CR
- T540-CR
- T420-CR
- T440-CR
- T422-CR
- T404-BT
- T440-LP-CR
- T420-LL-CR
- T420-CX

1.2. Software Requirements

1.2.1. Linux Requirements

Currently the iWARP driver is available for the following versions:

- RHEL 7.3, 3.10.0-514.el7
- RHEL 7.3, 3.10.0-514.el7.ppc64le (POWER8 LE)

- RHEL 7.3, 4.5.0-15.el7.aarch64 (ARM64)
- RHEL 7.2, 3.10.0-327.el7
- RHEL 6.8, 2.6.32-642.el6
- RHEL 6.7, 2.6.32-573.el6
- SLES 12 SP2, 4.4.21-69-default
- SLES 12 SP1, 3.12.49-11-default
- SLES 11 SP4, 3.0.101-63-default
- Ubuntu 16.04.1, 4.4.0-47-generic
- Ubuntu 14.04.4, 4.2.0-27-generic
- Kernel.org linux-4.8

Other kernel versions have not been tested and are not guaranteed to work

1. Software/Driver Loading

Important *Please ensure that all inbox drivers are unloaded before proceeding with unified wire drivers.*

1.1. Loading iWARP driver

The driver must be loaded by the root user. Any attempt to load the driver as a regular user will fail.

To load the iWARP driver we need to load the NIC driver and core RDMA drivers first. Run the following commands:

```
[root@host~]# modprobe cxgb4
[root@host~]# modprobe iw_cxgb4
[root@host~]# modprobe rdma_ucm
```

2. Software/Driver Unloading

To unload the iWARP driver, run the following command:

```
[root@host~]# rmmod iw_cxgb4
```

Important *openmpi-1.4.3 can cause IMB benchmark stalls due to a shared memory BTL issue. This issue is fixed in openmpi-1.4.5 and later releases. Hence, it is recommended that you download and install the latest stable release from Open MPI's official website,* http://www.open-mpi.org

3. Software/Driver Configuration and Fine-tuning

3.1. Testing connectivity with *ping* and *rping*

Load the NIC, iWARP & core RDMA modules as mentioned in Software/Driver Loading section. After which, you will see two or four ethernet interfaces for the Terminator device. Configure them with an appropriate ip address, netmask, etc. You can use the Linux *ping* command to test basic connectivity via the Terminator interface. To test RDMA, use the *rping* command that is included in the librdmacm-utils RPM:

Run the following command on the server machine:

```
[root@host~]# rping -s -a server_ip_addr -p 9999
```

Run the following command on the client machine:

```
[root@host~]# rping -c –Vv -C10 -a server_ip_addr -p 9999
```

You should see ping data like this on the client:

```
ping data: rdma-ping-0: ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqr
ping data: rdma-ping-1: BCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrs
ping data: rdma-ping-2: CDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrst
ping data: rdma-ping-3: DEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstu
ping data: rdma-ping-4: EFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuv
ping data: rdma-ping-5: FGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvw
ping data: rdma-ping-6: GHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwx
ping data: rdma-ping-7: HIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxy
ping data: rdma-ping-8: IJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz
ping data: rdma-ping-9: JKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyzA
client DISCONNECT EVENT...
#
```

3.2. Enabling various MPIs

3.2.1. Setting shell for Remote Login

User needs to set up authentication on the user account on all systems in the cluster to allow user to remotely logon or executing commands without password.

Quick steps to set up user authentication:

i. Change to user home directory

```
[root@host~]# cd
```

ii. Generate authentication key

```
[root@host~]# ssh-keygen -t rsa
```

iii. Hit [Enter] upon prompting to accept default setup and empty password phrase
iv. Create authorization file

```
[root@host~]# cd .ssh
[root@host~]# cat *.pub > authorized_keys
[root@host~]# chmod 600 authorized_keys
```

v. Copy directory .ssh to all systems in the cluster

```
[root@host~]# cd
[root@host~]# scp -r /root/.ssh remotehostname-or-ipaddress:
```

3.2.2. Configuration of various MPIs (Installation and Setup)

- **Intel-MPI**

i. Download latest Intel MPI from the Intel website
ii. Copy the license file (.lic file) into `l_mpi_p_x.y.z` directory
iii. Create machines.LINUX (list of node names) in `l_mpi_p_x.y.z`
iv. Select advanced options during installation and register the MPI.
v. Install software on every node.

```
[root@host~]# ./install.py
```

vi. Set IntelMPI with mpi-selector (do this on all nodes).

```
[root@host~]# mpi-selector --register intelmpi --source-dir
/opt/intel/impi/3.1/bin/
[root@host~]# mpi-selector --set intelmpi
```

vii. Edit `.bashrc` and add these lines:

```
export RSH=ssh
export DAPL_MAX_INLINE=64
export I_MPI_DEVICE=rdssm:chelsio
export MPIEXEC_TIMEOUT=180
export MPI_BIT_MODE=64
```

viii. Logout & log back in.

ix. Populate `mpd.hosts` with node names.

Note
- *The hosts in this file should be Chelsio interface IP addresses.*
- `I_MPI_DEVICE=rdssm:chelsio` *assumes you have an entry in* `/etc/dat.conf` *named* `chelsio`*.*
- `MPIEXEC_TIMEOUT` *value might be required to increase if heavy traffic is going across the systems.*

x. Contact Intel for obtaining their MPI with DAPL support.

xi. To run Intel MPI over RDMA interface, DAPL 2.0 should be set up as follows:

Enable the Chelsio device by adding an entry at the beginning of the /`etc/dat.conf` file for the Chelsio interface. For instance, if your Chelsio interface name is eth2, then the following line adds a DAT version 2.0 device named “chelsio2" for that interface:

```
chelsio2 u2.0 nonthreadsafe default libdaplofa.so.2 dapl.2.0 "eth2 0" ""
```

Open MPI (Installation and Setup)

Open MPI iWARP support is only available in Open MPI version 1.3 or greater.

Open MPI will work without any specific configuration via the openib btl. Users wishing to performance tune the configurable options may wish to inspect the receive queue values. Those can be found in the "Chelsio T4" section of `mca-btl-openib-device-params.ini`. Follow the steps mentioned below to install and configure Open MPI.

i. If not alreay done, install *mpi-selector* tool.

ii. Download the latest stable/feature version of openMPI from OpenMPI website, http://www.open-mpi.org
iii. Untar and change your current working directory to openMPI package directory.
iv. Configure and install as:

```
[root@host~]#./configure --with-openib=/usr  CC=gcc CXX=g++ F77=gfortran
FC=gfortran --enable-mpirun-prefix-by-default --prefix=/usr/mpi/gcc/openmpi-
x.y.z/ --with-openib-libdir=/usr/lib64/ --libdir=/usr/mpi/gcc/openmpi-
x.y.z/lib64/ --with-contrib-vt-flags=--disable-iotrace
[root@host~]# make
[root@host~]# make install
```

The above step will install openMPI in */usr/mpi/gcc/openmpi-x.y.z/*

Note *To enable multithreading, add "--enable-mpi-thread-multiple" and "--with-threads=posix" parameters to the above configure command.*

v. Next, create a shell script , *mpivars.csh,* with the following entry:

```
# path
if ("" == "`echo $path | grep /usr/mpi/gcc/openmpi-x.y.z/bin`") then
   set path=(/usr/mpi/gcc/openmpi-x.y.z/bin $path)
endif


# LD_LIBRARY_PATH
if ("1" == "$?LD_LIBRARY_PATH") then
    if ("$LD_LIBRARY_PATH" !~ */usr/mpi/gcc/openmpi-x.y.z/lib64*) then
    setenv LD_LIBRARY_PATH /usr/mpi/gcc/openmpi-
x.y.z/lib64:${LD_LIBRARY_PATH}
    endif
else
   setenv LD_LIBRARY_PATH /usr/mpi/gcc/openmpi-x.y.z/lib64
endif
# MPI_ROOT
setenv MPI_ROOT /usr/mpi/gcc/openmpi-x.y.z
```

vi. Simlarly, create another shell script, *mpivars.sh*, with the following entry:

```
# PATH
if test -z "`echo $PATH | grep /usr/mpi/gcc/openmpi-x.y.z/bin`"; then
    PATH=/usr/mpi/gcc/openmpi-x.y.z/bin:${PATH}
    export PATH
fi
# LD_LIBRARY_PATH
if test -z "`echo $LD_LIBRARY_PATH | grep /usr/mpi/gcc/openmpi-
   x.y.z/lib64`"; then
   LD_LIBRARY_PATH=/usr/mpi/gcc/openmpi-
   x.y.z/lib64${LD_LIBRARY_PATH:+:}${LD_LIBRARY_PATH}
   export LD_LIBRARY_PATH
fi


# MPI_ROOT
MPI_ROOT=/usr/mpi/gcc/openmpi-x.y.z
export MPI_ROOT
```

vii. Next, copy the two files created in steps (v) and (vi) to */usr/mpi/gcc/openmpi-x.y.z/bin* and */usr/mpi/gcc/openmpi-x.y.z/etc*

viii. Register OpenMPI with MPI-selector:

```
[root@host~]# mpi-selector --register openmpi --source-dir
/usr/mpi/gcc/openmpi-x.y.z/bin
```

ix. Verify if it is listed in mpi-selector:

```
[root@host~]# mpi-selector --l
```

x. Set OpenMPI:

```
[root@host~]# mpi-selector --set openmpi –yes
```

xi. Logut and log back in.

MVAPICH2 (Installation and Setup)

i. Download the latest MVAPICH2 software package from http://mvapich.cse.ohio-state.edu/
ii. Untar and change your current working directory to MVAPICH2 package directory.
iii. Configure and install as:

```
[root@host~]# ./configure --prefix=/usr/mpi/gcc/mvapich2-x.y/ --with-
device=ch3:mrail --with-rdma=gen2 --enable-shared --with-ib-
libpath=/usr/lib64/ -enable-rdma-cm --libdir=/usr/mpi/gcc/mvapich2-x.y/lib64
[root@host~]# make
[root@host~]# make install
```

The above step will install MVAPICH2 in */usr/mpi/gcc/mvapich2-x.y/*

iv. Next, create a shell script , mpivars.csh, with the following entry:

```
# path
if ("" == "`echo $path | grep /usr/mpi/gcc/mvapich2-x.y/bin`") then
   set path=(/usr/mpi/gcc/mvapich2-x.y/bin $path)
endif


# LD_LIBRARY_PATH
if ("1" == "$?LD_LIBRARY_PATH") then
    if ("$LD_LIBRARY_PATH" !~ */usr/mpi/gcc/mvapich2-x.y/lib64*) then
    setenv LD_LIBRARY_PATH /usr/mpi/gcc/mvapich2-
x.y/lib64:${LD_LIBRARY_PATH}
    endif
else
   setenv LD_LIBRARY_PATH /usr/mpi/gcc/mvapich2-x.y/lib64
endif


# MPI_ROOT
setenv MPI_ROOT /usr/mpi/gcc/mvapich2-x.y
```

v. Simlarly, create another shell script, *mpivars.sh*, with the following entry:

```
# PATH
if test -z "`echo $PATH | grep /usr/mpi/gcc/ mvapich2-x.y/bin`"; then
    PATH=/usr/mpi/gcc/mvapich2-x.y/bin:${PATH}
    export PATH
fi
# LD_LIBRARY_PATH
if test -z "`echo $LD_LIBRARY_PATH | grep /usr/mpi/gcc/mvapich2-
x.y/lib64`"; then
   LD_LIBRARY_PATH=/usr/mpi/gcc/mvapich2-
x.y/lib64${LD_LIBRARY_PATH:+:}${LD_LIBRARY_PATH}
   export LD_LIBRARY_PATH
fi


# MPI_ROOT
MPI_ROOT=/usr/mpi/gcc/mvapich2-x.y
export MPI_ROOT
```

vi. Next, copy the two files created in steps (iv) and (v) *to /usr/mpi/gcc/mvapich2-x.y/bin* and */usr/mpi/gcc/mvapich2-x.y/etc*

vii. Add the following entries in *.bashrc* file:

```
export MVAPICH2_HOME=/usr/mpi/gcc/mvapich2-x.y/
export MV2_USE_IWARP_MODE=1
export MV2_USE_RDMA_CM=1
```

viii. Register MPI:

```
[root@host~]# mpi-selector --register mvapich2 --source-dir
/usr/mpi/gcc/mvapich2-x.y/bin/
```

ix. Verify if it is listed in mpi-selector:

```
[root@host~]# mpi-selector --l
```

x. Set MVAPICH2:

```
[root@host~]# mpi-selector --set mvapich2 –yes
```

xi. Logut and log back in.
xii. Populate `mpd.hosts` with node names.
xiii. On each node, create /etc/mv2.conf with a single line containing the IP address of the local adapter interface. This is how MVAPICH2 picks which interface to use for RDMA traffic.

3.2.3. Building MPI tests

i. Download *Intel's MPI Benchmarks* from http://software.intel.com/en-us/articles/intel-mpi-benchmarks
ii. Untar and change your current working directory to *src* directory.
iii. Edit *make_mpich* file and set *MPI_HOME* variable to the MPI which you want to build the benchmarks tool against. For example, in case of openMPI-1.6.4 set the variable as:

```
MPI_HOME=/usr/mpi/gcc/openmpi-1.6.4/
```

iv. Next, build and install the benchmarks using:

```
[root@host~]# gmake -f make_mpich
```

The above step will install IMB-MPI1, IMB-IO and IMB-EXT benchmarks in the current working directory (i.e. *src*).

v. Change your working directory to the MPI installation directory. In case of OpenMPI, it will be */usr/mpi/gcc/openmpi-x.y.z/*
vi. Create a directory called *tests* and then another directory called *imb* under *tests.*
vii. Copy the benchmarks built and installed in step (iv) to the *imb* directory.
viii. Follow steps (v), (vi) and (vii) for all the nodes.

3.2.4. Running MPI applications

- Run Intel MPI applications as:

```
mpdboot -n <no_of_nodes_in_cluster> -r ssh
mpdtrace
mpiexec -ppn -n 2 /opt/intel/impi/3.1/tests/IMB-3.1/IMB-MPI1
```

The performance is best with NIC MTU set to 9000 bytes.

- Run Open MPI application as:

```
mpirun --host  node1,node2 -mca btl openib,sm,self /usr/mpi/gcc/openmpi-
x.y.z/tests/imb/IMB-MPI1
```

Note *For OpenMPI/RDMA clusters with node counts greater than or equal to 8 nodes, and process counts greater than or equal to 64, you may experience the following RDMA address resolution error when running MPI jobs with the default OpenMPI settings:*

```
The RDMA CM returned an event error while attempting to make a connection.
This type of error usually indicates a network configuration error.
Local host:   core96n3.asicdesigners.com
Local device: Unknown
Error name:   RDMA_CM_EVENT_ADDR_ERROR
Peer:         core96n8
```

Workaround*: Increase the OpenMPI rdma route resolution timeout. The default is 1000, or 1000ms. Increase it to 30000 with this parameter:*

```
--mca btl_openib_connect_rdmacm_resolve_timeout 30000
```

- Run MVAPICH2 application as :

```
mpirun_rsh -ssh -np 8 -hostfile mpd.hosts $MVAPICH2_HOME/tests/imb/IMB-MPI1
```

3.3. Setting up NFS-RDMA

3.3.1. Starting NFS-RDMA

- **Server-side settings**

Follow the steps mentioned below to set up an NFS-RDMA server.

i. Make entry in `/etc/exports` file for the directories you need to export using NFS-RDMA on server as:

```
/share/rdma      *(fsid=0,async,insecure,no_root_squash)
/share/rdma1     *(fsid=1,async,insecure,no_root_squash)
```

Note that for each directory you export, you should have DIFFERENT fsid's.

ii. Load the iwarp modules and make sure peer2peer is set to 1.
iii. Load `xprtrdma` and `svcrdma` modules as:

```
[root@host~]# modprobe xprtrdma
[root@host~]# modprobe svcrdma
```

iv. Start the nfs service as:

```
[root@host~]# service nfs start
```

All services in NFS should start without errors.

v. Now we need to edit the file portlist in the path `/proc/fs/nfsd/`
Include the rdma port 2050 into this file as:

```
[root@host~]# echo rdma 2050 > /proc/fs/nfsd/portlist
```

vi. Run `exportfs` to make local directories available for Network File System (NFS) clients to mount.

```
[root@host~]# exportfs
```

Now the NFS-RDMA server is ready.

- **Client-side settings**

Follow the steps mentioned below at the client side.

i. Load the iwarp modules and make sure peer2peer is set to 1. Make sure you are able to ping and ssh to the server Chelsio interface through which directories will be exported.

ii. Load the `xprtrdma` module.

```
[root@host~]# modprobe xprtrdma
```

iii. Run the `showmount` command to show all directories from server as:

```
[root@host~]# showmount -e <server-chelsio-ip>
```

iv. Once the exported directories are listed, mount them as:

```
[root@host~]# mount.nfs <serverip>:<directory> <mountpoint-on-client> -o
vers=3,rdma,port=2050,wsize=65536,rsize=65536
```

3.4. Performance Tuning

See the **Performance Tuning** section in the **Unified Wire** chapter for generic performance settings.

V. iSER

4. Introduction

The iSCSI Extensions for RDMA (iSER) protocol is a translation layer for operating iSCSI over RDMA transports, such as iWARP/Ethernet or InfiniBand.

4.1. Hardware Requirements

4.1.1. Supported adapters

The following are the currently shipping Chelsio adapters that are compatible with Chelsio iSER driver:

- T62100-LP-CR
- T6225-CR
- T520-BT
- T580-CR
- T580-LP-CR
- T520-LL-CR
- T520-CR
- T540-CR

4.2. Software Requirements

4.2.1. Linux Requirements

Currently the iSER driver is available for the following version(s):

- Kernel.org linux-4.8 (kernel compiled on RHEL 7.X distribution)

Other kernel versions have not been tested and are not guaranteed to work

5. Software/Driver Loading

Follow the steps mentioned below on both target and initiator machines:

i. Unload Chelsio iWARP driver if previously loaded:

```
[root@host~]# rmmod iw_cxgb4
```

ii. Load the following modules:

```
[root@host~]# modprobe iw_cxgb4 mpa_rev=2
[root@host~]# modprobe rdma_ucm
```

iii. Start the iWARP Port Mapper Daemon:

```
[root@host~]# iwpmd
```

iv. Bring up the Chelsio interface(s):

```
[root@host~]# ifconfig ethX x.x.x.x up
```

v. Run the following script for IRQ mapping:

```
[root@host~]# t4_perftune.sh
```

vi. On target, run the following command:

```
[root@host~]# modprobe ib_isert
```

On initiator, run the following command:

```
[root@host~]# modprobe ib_iser
```

6. Software/Driver Configuration and Fine-tuning

6.1. Configuring LIO Target

The following commands will configure LIO target with iSER support, using ramdisk as LUN:

```
[root@host~]# targetcli /backstores/ramdisk create name=ram0 size=1GB
[root@host~]# targetcli /iscsi create wwn=iqn.2003-01.org.lun0.target
[root@host~]# targetcli /iscsi/iqn.2003-01.org.lun0.target/tpg1/luns create
/backstores/ramdisk/ram0
[root@host~]# targetcli /iscsi/iqn.2003-
01.org.lun0.target/tpg1/portals/0.0.0.0:3260 enable_iser boolean=True
[root@host~]# targetcli /iscsi/iqn.2003-01.org.lun0.target/tpg1 set
attribute authentication=0 demo_mode_write_protect=0 generate_node_acls=1
cache_dynamic_acls=1
[root@host~]# targetcli saveconfig
```

6.2. Configuring OpeniSCSI Initiator

The following commands will log-in to the LIO target using OpeniSCSI initiator, with iSER as transport mode:

```
[root@host~]# iscsiadm -m discovery -t st -p  102.10.10.4
[root@host~]# iscsiadm -m node -p 102.10.10.4 -T iqn.2003-01.org.lun0.target
--op update -n node.transport_name -v iser
[root@host~]# iscsiadm -m node -p 102.10.10.4 -T iqn.2003-01.org.lun0.target
--login
```

VI. WD-UDP

1. Introduction

Chelsio WD-UDP (Wire Direct-User Datagram Protocol) with Multicast is a user-space UDP stack with Multicast address reception and socket acceleration that enables users to run their existing UDP socket applications unmodified.

It features software modules that enable direct wire access from user space to the Chelsio network adapter with complete bypass of the kernel, which results in an ultra-low latency Ethernet solution for high frequency trading and other delay-sensitive applications.

1.1. Hardware Requirements

1.1.1. Supported adapters

The following are the currently shipping Chelsio adapters that are compatible with Chelsio WD-UDP driver:

- T62100-LP-CR
- T6225-CR
- T520-BT
- T580-CR
- T520-LL-CR
- T520-CR
- T580-LP-CR
- T540-CR
- T420-CR
- T440-CR
- T422-CR
- T404-BT
- T440-LP-CR
- T420-LL-CR
- T420-CX

1.2. Software Requirements

1.2.1. Linux Requirements

Currently the WD-UDP driver is available for the following versions:

- RHEL 7.3, 3.10.0-514.el7
- RHEL 7.2, 3.10.0-327.el7
- RHEL 6.8, 2.6.32-642.el6
- RHEL 6.7, 2.6.32-573.el6
- SLES 12 SP2, 4.4.21-69-default

- SLES 12 SP1, 3.12.49-11-default
- SLES 11 SP4, 3.0.101-63-default
- Ubuntu 16.04.1, 4.4.0-47-generic
- Ubuntu 14.04.4, 4.2.0-27-generic
- Kernel.org linux-4.8

Other kernel versions have not been tested and are not guaranteed to work.

2. Software/Driver Loading

Important *Please ensure that all inbox drivers are unloaded before proceeding with unified wire drivers.*

The driver must be loaded by the root user. Any attempt to load the driver as a regular user will fail.

RDMA core modules from the OFED package should be loaded before proceeding. To load the WD-UDP driver, use the following commands which will automatically load RDMA core modules:

```
[root@host~]# modprobe cxgb4
[root@host~]# modprobe iw_cxgb4
[root@host~]# modprobe rdma_ucm
```

3. Software/Driver Unloading

To unload the WD-UDP driver, run the following command:

```
[root@host~]# rmmod iw_cxgb4
```

4. Software/Driver Configuration and Fine-tuning

4.1. Accelerating UDP Socket communications

The *libcxgb4_sock* library is a LD_PRELOAD-able library that accelerates UDP Socket communications transparently and without recompilation of the user application. This section describes how to use libcxgb4_sock.

By preloading *libcxgb4_sock*, all sockets created by the application are intercepted and possibly accelerated based on the user's configuration. Once accelerated, data for the UDP endpoint are transmitted or received via HW queues allocated specifically for the accelerated endpoint, bypassing the kernel, the host networking stack and sockets framework, and enabling ultra-low latency and high bandwidth utilization.

Due to HW resource limitations, only a small number of queues can be allocated for UDP acceleration. Therefore, only performance critical UDP applications should use *libcxgb4_sock*.

Only 64 IPv4 UDP / 28 IPv6 UDP sockets can be accelerated per Chelsio device, with *Unified Wire Configuration* tuning option. If you want more sockets to be accelerated, please use *Low Latency* or *High Capacity WD* tuning option.

4.1.1. Application Requirements

Certain application behavior is not supported by *libcxb4_sock* in this release. If your application does any of the following, it will not work with *libcxgb4_sock*:

- Calling fork() after creating UDP sockets and using the UDP socket in the child process.
- Using multiple threads on a single UDP socket without serialization. For instance, having one thread sending concurrently with another thread receiving. If your application does this, you need to serialize these paths with a spin or mutex lock.
- Only 1 UDP endpoint is allowed to bind to a given port per host. So if you have multiple processes on the same host binding to the same UDP port number, you cannot use *libcxgb4_sock*.
- Applications must have root privileges to use *libcxgb4_sock*.
- Applications requiring bonded adapter interfaces are not currently supported.

The performance benefit observed with *libcxgb4_sock* will vary based on your application's behavior. While all UDP I/O is handled properly, only certain datagrams are accelerated. Non accelerated I/O is handled by *libcxgb4_sock* via the host networking stack seamlessly. Both Unicast and Multicast datagrams can be accelerated, but the datagrams must meet the following criteria:

- Non fragmented. In other words, they fit in a single IP datagram that is <= the adapter device MTU.

- Routed through the Terminator acceleration device. If the ingress datagram arrives via a device other than the Terminator acceleration device, then it will not utilize the acceleration path. On egress, if the destination IP address will not route out via the Terminator device, then it too will not be accelerated.

4.1.2. Using *libcxgb4_sock*

The *libcxgb4_sock* library utilizes the Linux RDMA Verbs subsystem, and thus requires the RDMA modules be loaded. Ensure that your systems load the *iw_cxgb4* and *rdma_ucm* modules:

```
[root@host~]# modprobe iw_cxgb4
[root@host~]# modprobe rdma_ucm
```

Now, preload *libcxgb4_sock*, using one of the methods mentioned below when starting your application:

- **Preloading using *wdload* script**:

```
[root@host~]# PROT=UDP wdload <pathto>/your_application
```

The above command will generate an end point file, *libcxgb4_sock.conf* at /etc/. Parameters like interface name and port number can be changed in this file.

Note *If you encounter error while using wdload on kernels built on RHEL 5.x distribution, then run the above command as :*
[root@host~]# NUMA=0 PROT=UDP wdload <pathto>/your_application

- **Preloading manually**

Create a configuration file that defines which UDP endpoints should be accelerated, their vlan and priority if any, as well as which Terminator interface/port should be used. The file `/etc/libcxgb4_sock.conf` contains these endpoint entries. Create this file on all systems using *libcxgb4_sock*. Here is the syntax:

```
#
# Syntax:
#
# endpoint {attributes} ...
# where attributes include:
#                 interface = interface-name
#                 port = udp-port-number
#                 vlan = vlan-id
#                 priority = vlan-priority
```

```
#
# e.g.
# endpoint {
#                interface=eth2.5
#                port = 8000 vlan = 5 priority=1
#    }
# endpoint { interface=eth2 port=9999}
#
# endpoints that bind to port 0 (requesting the host allocate a port)

# can be accelerated with port=0:
#
# endpoint {interface=eth1 port=0}
#
```

Assume your Terminator interface is eth2. To accelerate all applications that preload *libcxgb4_sock* using eth2, you only need one entry in `/etc/libcxgb4_sock.conf`:

```
endpoint {interface=eth2 port=0}
```

If you have eth2 and eth3 configured for example, you can define certain endpoints to eth2 and others to eth3:

```
endpoint {interface=eth2 port=9999}
endpoint {interface=eth3 port=8888}
```

For VLAN support, create your VLANs using the normal OS service (like vconfig, for example), then add entries to define the VLAN and priority for each endpoint to be accelerated:

```
endpoint {interface = eth2.5 port=10000}
endpoint {interface = eth2.7 priority=3 port=9000}
```

Now, preload *libcxgb4_sock*:

```
[root@host~]# CXGB4_SOCK_CFG=<path to config file>
LD_PRELOAD=libcxgb4_sock.so <pathto>/your_application
```

Note *In order to offload IPv6 UDP sockets, please select "low latency networking" as configuration tuning option during installation.*

4.1.3. Running WD-UDP in debug mode

To use *libcxgb4_sock*'s debug capabilities, use the *libcxgb4_sock_debug* library provided in the package. Follow the steps mentioned below:

i. Make the following entry in the */etc/syslog.conf* file:

```
*.debug                              /var/log/cxgb4.log
```

ii. Restart the service:

```
[root@host~]# /etc/init.d/syslog restart
```

iii. Finally, preload *libcxgb4_sock_debug* using the command mentioned below when starting your application:

```
[root@host~]# LD_PRELOAD=libcxgb4_sock_debug.so CXGB4_SOCK_DEBUG=-1
<pathto>/your_application
```

4.1.4. Running WD-UDP with larger I/O size

If the I/O size is > 3988, execute the commands mentioned below:

```
[root@host~]# echo 1024 > /proc/sys/vm/nr_hugepages
[root@host~]# CXGB4_SOCK_HUGE_PAGES=1 PROT=UDP wdload
<pathto>/your_application
```

4.1.5. Example with hpcbench/udp

The udp benchmark from the hpcbench suite can be used to show the benefits of libcxgb4_sock. The hpcbench suite can be found at:

Source: http://hpcbench.sourceforge.net/index.html

Sample: http://hpcbench.sourceforge.net/udp.html

The nodes in this example, r9 and r10, have Terminator eth1 configured and the ports are connected point-to-point.

```
[root@r9 ~]# ifconfig eth1|grep inet
          inet addr:192.168.2.111  Bcast:192.168.2.255  Mask:255.255.255.0
          inet6 addr: fe80::7:4300:104:465a/64 Scope:Link

[root@r9 ~]#
[root@r10 ~]# ifconfig eth1|grep inet
          inet addr:192.168.2.112  Bcast:192.168.2.255  Mask:255.255.255.0
          inet6 addr: fe80::7:4300:104:456a/64 Scope:Link
[root@r10 ~]#
```

For this benchmark, we need a simple "accelerate all" configuration on both nodes:

```
[root@r9 ~]# cat /etc/libcxgb4_sock.conf
endpoint {interface=eth1 port=0}
[root@r9 ~]#

[root@r10 ~]# cat /etc/libcxgb4_sock.conf
endpoint {interface=eth1 port=0}
[root@r10 ~]#
```

On R10, we run udpserver on port 9000 without *libcxgb4_sock* preloaded, and on port 90001 with preload:

```
[root@r10 ~]# /usr/local/src/hpcbench/udp/udpserver -p 9000 &
[1] 11453
[root@r10 ~]# TCP socket listening on port [9000]

[root@r10 ~]# LD_PRELOAD=libcxgb4_sock.so
/usr/local/src/hpcbench/udp/udpserver -p 9001 &
[2] 11454
[root@r10 ~]# TCP socket listening on port [9001]
[root@r10 ~]#
```

Then on r9, we run udptest to port 9000 to see the host stack UDP latency:

```
[root@r9 ~]# /usr/local/src/hpcbench/udp/udptest -r 5 -a -h 192.168.1.112 -p
9000
```

Running the same test with *libcxgb4_sock*:

```
[root@r9 ~]# LD_PRELOAD=libcxgb4_sock.so /usr/local/src/hpcbench/udp/udptest
-r 5 -a -h 192.168.1.112 -p 9001
```

4.1.6. Determining if the application is being offloaded

To see if the application is being offloaded, open a window on one of the machines, and run `tcpdump` against the Chelsio interface. If you see minimal UDP output on the interface, then the UDP traffic is being properly offloaded.

VII. NVMe-oF

1. Introduction

NVMe over Fabrics specification extends the benefits of NVMe to large fabrics, beyond the reach and scalability of PCIe. NVMe enables deployments with hundreds or thousands of SSDs using a network interconnect, such as RDMA over Ethernet. Thanks to an optimized protocol stack, an end-to-end NVMe solution is expected to reduce access latency and improve performance, particularly when paired with a low latency, high efficiency transport such as RDMA. This allows applications to achieve fast storage response times, irrespective of whetherthe NVMe SSDs are attached locally or accessed remotely across enterprise or datacenter networks.

1.1. Hardware Requirements

1.1.1. Supported adapters

The following are the currently shipping Chelsio adapters that are compatible with Chelsio NVMe-oF driver:

- T62100-LP-CR
- T6225-CR
- T580-CR
- T580-LP-CR
- T540-CR
- T520-CR
- T520-LL-CR
- T520-BT

1.2. Software Requirements

1.2.1. Linux Requirements

Currently the NVMe-oF driver is available for the following version(s):

- Kernel.org linux-4.8

Other kernel versions have not been tested and are not guaranteed to work.

2. Software/Driver Loading

Important *Ensure that the kernel is installed with NVMe-oF components (refer Pre-requisites section) before proceeding with Unified Wire installation.*

Follow the steps mentioned below on both target and initiator machines:

i. Load the following modules:

```
[root@host~]# modprobe iw_cxgb4
[root@host~]# modprobe rdma_ucm
```

ii. Bring up the Chelsio interface(s):

```
[root@host~]# ifconfig ethX x.x.x.x up
```

iii. Mount configfs by running the below command:

```
[root@host~]# mount -t configfs none /sys/kernel/config
```

iv. On target, run the following commands:

```
[root@host~]# modprobe null_blk
[root@host~]# modprobe nvmet
[root@host~]# modprobe nvmet-rdma
```

On initiator, run the following commands:

```
[root@host~]# modprobe nvme
[root@host~]# modprobe nvme-rdma
```

3. Software/Driver Configuration and Fine-tuning

The following sections describe the method to configure target and initiator:

3.1. Target

i. The following commands will configure target using *nvmetcli* with a LUN:

```
[root@host~]# nvmetcli
/> cd subsystems
/subsystems> create nvme-ram0
/subsystems> cd nvme-ram0/namespaces
/subsystems/n...m0/namespaces> create nsid=1
/subsystems/n...m0/namespaces> cd 1
/subsystems/n.../namespaces/1> set device path=/dev/ram1
/subsystems/n.../namespaces/1> cd ../..
/subsystems/nvme-ram0> set attr allow_any_host=1
/subsystems/nvme-ram0> cd namespaces/1
/subsystems/n.../namespaces/1> enable
/subsystems/n.../namespaces/1> cd ../../../..
/> cd ports
/ports> create 1
/ports> cd 1/
/ports/1> set addr adrfam=ipv4.
/ports/1> set addr trtype=rdma
/ports/1> set addr trsvcid=4420
/ports/1> set addr traddr=102.1.1.102
/ports/1> cd subsystems
/ports/1/subsystems> create nvme-ram0
```

ii. Save the target configuration to a file:

```
/ports/1/subsystems> saveconfig /root/nvme-target_setup
/ports/1/subsystems> exit
```

iii. Clear the targets:

```
[root@host~]# nvmetcli clear
```

3.2. Initiator

i. Discover the target

```
[root@host~]# nvme discover -t rdma -a <target_ip> -s 4420
```

ii. Connect to target

- Connecting to a specific target:

```
[root@host~]# nvme connect -t rdma -a <target_ip> -s 4420 -n <target_name>
```

- Connecting to all targets configured on a portal

```
[root@host~]# nvme connect-all -t rdma -a <target_ip> -s 4420
```

iii. List the connected targets

```
[root@host~]# nvme list
```

iv. Format and mount the NVMe disks shown with the above command.

v. Disconnect from the target and unmount the disk:

```
[root@host~]# nvme disconnect -d <nvme_disk_name>
```

Note *nvme_disk_name is the name of the device (E.g.: nvme0n1) and not the device path.*

4. Software/Driver Unloading

Follow the steps mentioned below to unload the drivers:

On target, run the following commands:

```
[root@host~]# rmmod nvmet-rdma
[root@host~]# rmmod nvmet
[root@host~]# rmmod iw_cxgb4
```

On initiator, run the following commands:

```
[root@host~]# rmmod nvme-rdma
[root@host~]# rmmod nvme
[root@host~]# rmmod iw_cxgb4
```

VIII. LIO iSCSI Target Offload

1. Introduction

Linux-IO Target (LIO) is the in-kernel SCSI target implementation in Linux. This open-source standard supports common storage fabrics, including Fibre Channel, FCoE, iEEE 1394, iSCSI, NVMe-oF, iSER, SRP, USB, vHost, etc. The LIO iSCSI fabric module implements many advanced iSCSI features that increase performance and resiliency.

The LIO iSCSI Target Offload driver provides the following high level features:

- Offloads TCP/IP.
- Offloads iSCSI Header and Data Digest Calculations.
- Offload Speeds at 10Gb and 40Gb.
- Supports Direct Data Placement (DDP).
- Supports iSCSI Segmentation Offload.
- Supports iSCSI PDU recovery.

1.1. Hardware Requirements

1.1.1. Supported adapters

The following are the currently shipping Chelsio adapters that are compatible with Chelsio LIO iSCSI Target Offload driver:

- T62100-LP-CR
- T6225-CR
- T580-CR
- T580-LP-CR
- T540-CR
- T520-LL-CR
- T520-CR
- T520-BT

1.2. Software Requirements

cxgb4, iscsi_target_mod, target_core_mod, ipv6 modules are required by LIO iSCSI Target Offload (`cxgbit.ko`) module to work.

1.2.1. Linux Requirements

Currently the LIO iSCSI Target Offload driver is available for the following version(s):

- Kernel.org linux-4.8

Other versions have not been tested and are not guaranteed to work.

1.2.2. iSCSI Requirements

The LIO iSCSI Target Offload driver requires the following components to be installed to function:

- Python* (v2.7.10 provided in the package)
- TargetCLI* (v2.1 provided in the package)
- OpenSSL (Download from https://www.openssl.org/source/)

* If not already present in the system, the component provided in the package will be installed along with the kernel.

2. Software/Driver Installation

2.1. Installation

i. Download the stable version of 4.8.X from kernel.org.
ii. Untar the tar-ball.
iii. Change your working directory to kernel package directory and run the following command to invoke the installation menu.

```
[root@host~]# make menuconfig
```

iv. Select **Device Drivers** > **Generic Target Core Mod (TCM) and ConfigFS Infrastructure**.
v. Enable **Linux-iSCSI.org iSCSI Target Mode Stack** and **Chelsio iSCSI target offload driver**.
vi. Select **Save**.
vii. Exit from the installation menu.
viii. Apply the patch provided in the Unified Wire package

```
[root@host~]# patch -p1 <
/root/<driver_package>/src/cxgbit/patch/iscsi_target.patch
```

ix. Continue with installation as usual.
x. Reboot to the newly installed 4.8 kernel. Verify by running *uname -a* command. You should see a similar output:

```
Linux decagon.asicdesigners.com 4.8.8 #1 SMP Wed Nov 23
21:47:06 IST 2016 x86_64 x86_64 x86_64 GNU/Linux
```

xi. Install the Unified Wire package as mentioned in the Chelsio Unified Wire chapter.

3. Software/Driver Loading

Important *Please ensure that all inbox drivers are unloaded before proceeding with unified wire drivers.*

The driver must be loaded by the root user. Any attempt to load the driver as a regular user will fail.

3.1. Loading in NIC mode (without full offload support)

To load the Network driver without full offload support, run the following command:

```
[root@host~]# modprobe cxgb4
```

Bring up the interface.

```
[root@host~]# ifconfig ethX <IP address> up
```

3.2. Loading LIO iSCSI Target Offload driver

Load the LIO iSCSI Target Offload driver (*cxgbit)* using:

```
[root@host~]# modprobe cxgbit
```

4. Software/Driver Configuration and Fine-tuning

4.1. Configuring LIO iSCSI Target

The LIO iSCSI Target needs to be configured before it can become useful. Please refer the user manual at http://www.linux-iscsi.org/Doc/LIO Admin Manual.pdf to do so.

4.1.1. Sample Configuration

Here is a sample iSCSI configuration listing a target configured with 1 RAM disk LUN and ACL not configured:

```
[root@host~]# targetcli ls
o- / ........................................................[...]
 o- backstores...................................................[...]
 | o- block..........................................[Storage Objects: 0]
 | o- fileio .....................................[Storage Objects: 0]
 | o- pscsi ......................................[Storage Objects: 0]
 | o- ramdisk .........................................[Storage Objects: 1]
 | | o- ramdisk01 ...................................[(300.0MiB) activated]
 | o- user .......................................[Storage Objects: 0]
 o- iscsi ..........................................[Targets: 1]
| o- iqn.2015-12.org.linux-iscsi.proton.target01..................[TPGs: 1]
|   o- tpg1 .........................................[gen-acls, no-auth]
|   o- acls ...............................................[ACLs: 0]
|   o- luns ...............................................[LUNs: 1]
|   | o- lun1 ......................................[ramdisk/ramdisk01]
|   o- portals .........................................[Portals: 1]
|    o- 102.10.10.121:3260 .....................................[OK]
o- loopback ...............................................[Targets: 0]
o- vhost ..............................................[Targets: 0]
```

4.2. Offloading LIO iSCSI Connection

To offload the LIO iSCSI Target use the following command:

```
[root@host~]# echo 1 >
/sys/kernel/config/target/iscsi/<target_iqn>/tpgt_1/np/<target_ip>\:3260/cxg
bit
```

E.g.:

```
[root@host~]# echo 1 > /sys/kernel/config/target/iscsi/iqn.2015-
12.org.linux-iscsi.proton.target01/tpgt_1/np/102.20.20.241\:3260/cxgbit
```

Execute the above command for every portal address listing on Chelsio interface.

4.3. Performance Tuning

In order to auto tune the system for best performance, Chelsio recommends:

- Disabling virtualization, c-state technology, VT-d, Intel I/O AT and SR-IOV in the BIOS settings.
- Installing the adapter into a PCIe Gen3 x8/x16 slot.
- Disable SELinux and firewall.
- Installing the driver package will copy `t4_perftune.sh` script to **/sbin** directory. Run the script to map the adapter queues to different CPUs:

```
[root@host~]# t4_perftune.sh
```

Also, follow the steps mentioned below to lower your latency:

i. Run the following script to disable few services.

```
[root@host~]# t4_latencytune.sh <interface>
```

ii. Set sysctl param *net.ipv4.tcp_low_latency* to 1

```
[root@host~]# sysctl -w net.ipv4.tcp_low_latency=1
```

5. Software/Driver Unloading

5.1. Unloading the LIO iSCSI Target Offload driver

To unload the LIO iSCSI Target Offload kernel module, follow the steps mentioned below:

i. Log out from the initiator.
ii. Run the following command:

```
[root@host~]# echo 0 >
/sys/kernel/config/target/iscsi/<target_iqn>/tpgt_1/np/<target_ip>\:3260/cxg
bit
```

Execute the above command for every portal address listening on Chelsio interface.

iii. Unload the driver:

```
[root@host~]# rmmod cxgbit
```

5.2. Unloading the NIC driver

To unload the NIC driver, run the following command:

```
[root@host~]# rmmod cxgb4
```

IX. iSCSI PDU Offload Target

1. Introduction

This section describes how to install and configure iSCSI PDU Offload Target software for use as a key element in your iSCSI SAN. The software runs on Linux-based systems that use Chelsio or non-Chelsio based Ethernet adapters. However, to guarantee highest performance, Chelsio recommends using Chelsio adapters. Chelsio's adapters include offerings that range from stateless offload adapters (regular NIC) to the full line of TCP/IP Offload Engine (TOE) adapters.

The software implements RFC 3720, the iSCSI standard of the IETF. The software has been fully tested for compliance to that RFC and others and it has been exhaustively tested for interoperability with the major iSCSI vendors.

The software implements most of the iSCSI protocol in software running in kernel mode on the host with the remaining portion, which consists of the entire fast data path, in hardware when used with Chelsio's TOE adapters. When standard NIC adapters are used the entire iSCSI protocol is executed in software.

The performance of this iSCSI stack is outstanding and when used with Chelsio's hardware it is enhanced further. Because of the tight integration with Chelsio's TOE adapters, this software has a distinct performance advantage over the regular NIC. The entire solution, which includes this software, Chelsio TOE hardware, an appropriate base computer system – including a high end disk subsystem, has industry leading performance. This can be seen when the entire solution is compared to others based on other technologies currently available on the market in terms of throughput and IOPS.

1.1. Features

Chelsio's iSCSI driver stack supports the iSCSI protocol in the Target mode. From henceforth "iSCSI Software Entity" term refers to the iSCSI target.

The Chelsio iSCSI PDU Offload Target software provides the following high level features:

- Expanded NIC Support
    - Chelsio TCP Offload Engine (TOE) Support
        - T6/T5/T4 Based HBAs (T6/T5/T4xx Series cards)
    - Non-Chelsio
        - Runs on regular NICs
- Chelsio Terminator ASIC Support
    - Offloads iSCSI Fast Data Path with Direct Data Placement (DDP)
    - Offloads iSCSI Header and Data Digest Calculations
    - Offload Speeds at 1Gb, 10Gb, 25Gb, 40Gb and 100Gb
    - Offloads TCP/IP for NAS simultaneously with iSCSI
- Target Specific features

- Full compliance with RFC 3720
- Error Recovery Level 0 (ERL 0)
- CHAP support for both discovery and login including mutual authentication
- Internet Storage Name Service (iSNS) Client
- Target Access Control List (ACL)
- Multiple Connections per Session
- Multiple Targets
- Multiple LUNs per Target
- Multi Path I/O (MPIO)
- Greater than 2 TB Disk Support
- Reserve / Release for Microsoft Cluster© Support
- Persistent Reservation
- Dynamic LUN Resizing
- iSCSI Target Redirection
- Multiple Target device types
    - Block
    - Virtual Block (LVM, Software RAID, EVMS, etc.)
    - Built in RAM Disk
    - Built in zero copy RAM Disk
- Supports iSCSI Boot Initiators
- An Intuitive and Feature Rich Management CLI

This chapter will cover these features in detail.

1.2. Hardware Requirements

1.2.1. Supported adapters

The following are the currently shipping Chelsio adapters that are compatible with iSCSI PDU Offload Target software:

- T62100-LP-CR
- T6225-CR
- T520-BT
- T580-CR
- T520-LL-CR
- T520-CR
- T580-LP-CR
- T540-CR
- T420-CR
- T440-CR
- T422-CR
- T404-BT

- T420-BCH
- T440-LP-CR
- T420-BT
- T420-LL-CR
- T420-CX

1.2.2. adapter Requirements

The Chelsio iSCSI PDU Offload Target software can be used with or without hardware protocol offload technology. There are two modes of operation using the iSCSI PDU Offload Target software on Ethernet-based adapters:

- Regular NIC – The software can be used in non-offloaded (regular NIC) mode. Please note however that this is the least optimal mode of operating the software in terms of performance.
- iSCSI HW Acceleration – In addition to offloading the TCP/IP protocols in hardware (TOE), this mode also takes advantage of Chelsio's ASIC capability of hardware assisted iSCSI data and header digest calculations as well as using the direct data placement (DDP) feature.

1.2.3. Storage Requirements

When using the Chelsio iSCSI target, a minimum of one hardware storage device is required. This device can be any of the device types that are supported (block, virtual block, RAM disk). Multiple storage devices are allowed by configuring the devices to one target or the devices to multiple targets. The software allows multiple targets to share the same device but use caution when doing this.

Chelsio's implementation of the target iSCSI stack has flexibility to accommodate a large range of configurations. For quick testing, using a RAM Disk as the block storage device works nicely. For deployment in a production environment a more sophisticated system would be needed. That typically consists of a system with one or more storage controllers with multiple disk drives attached running software or hardware based RAID.

1.3. Software Requirements

`chiscsi_base.ko` is iSCSI non-offload target mode driver and `chiscsi_t4.ko` is iSCSI PDU offload target mode driver.

cxgb4, toecore, t4_tom and chiscsi_base modules are required by `chiscsi_t4.ko` module to work in offloaded mode. Whereas in iscsi non-offloaded target (NIC) mode, only `cxgb4` is needed by `chiscsi_base.ko` module.

1.3.1. Linux Requirements

Currently the iSCSI PDU Offload Target software is available for the following versions:

- RHEL 7.3, 3.10.0-514.el7
- RHEL 7.3, 3.10.0-514.el7.ppc64le (POWER8 LE)
- RHEL 7.3, 4.5.0-15.el7.aarch64 (ARM64)
- RHEL 7.2, 3.10.0-327.el7
- RHEL 6.8, 2.6.32-642.el6
- RHEL 6.7, 2.6.32-573.el6
- SLES 12 SP2, 4.4.21-69-default
- SLES 12 SP1, 3.12.49-11-default
- SLES 11 SP4, 3.0.101-63-default
- Ubuntu 16.04.1, 4.4.0-47-generic
- Ubuntu 14.04.4, 4.2.0-27-generic
- Kernel.org linux-4.8

Other kernel versions have not been tested and are not guaranteed to work.

1.3.2. Requirements for Installing the iSCSI Software

When installing the iSCSI software, it is required that the system have Linux kernel source or its headers installed in order to compile the iSCSI software as a kernel module. The source tree may be only header files, as for RHEL6 as an example, or a complete tree. The source tree needs to be configured and the header files need to be compiled. Additionally, the Linux kernel must be configured to use modules.

2. Software/Driver Loading

Important *Please ensure that all inbox drivers are unloaded before proceeding with unified wire drivers.*

There are two main steps to installing the Chelsio iSCSI PDU Offload Target software. They are:

1. **Installing the iSCSI software** – The majority of this section deals with how to install the iSCSI software.
2. **Configuring the iSCSI software** – Information on configuring the software can be found in a section further into this user's guide.

2.1. Latest iSCSI Software Stack Driver Software

The iSCSI software stack comes bundled in the Chelsio Unified Wire package which can be downloaded from the Chelsio support website (http://service.chelsio.com).

The iSCSI software is available for use with most installations of the Linux kernel. The software is dependent on the underlying NIC adapter driver and thus the limitation on what version of the Linux kernel it can run on is mostly dependent on the NIC driver's limitations.

The iSCSI module will be installed in the /lib/modules/<linux_kernel_version>/updates/kernel/drivers/scsi/chiscsi directory. The modules database will be updated by the installer. This allows the iSCSI module to be located when using the `modprobe` utility. The actual module `chiscsi_t4.ko` can be found inside the package under /build/src/chiscsi/t4.

The `iscsictl` tool and the `chisns` tool will be installed in `/sbin`. The `chisns` tool starts the iSNS client. The `iscsictl` tool is provided for configuring and managing the iSCSI targets and iSNS client. It also provides control for iSCSI global settings.

1. Loading the Kernel module

- Run modprobe as follows:

```
[root@host~]# modprobe chiscsi_t4
```

Note *i. While using rpm-tar-ball for installation*

a. Uninstallation will result into chiscsi.conf file renamed into chiscsi.conf.rpmsave, but if again uninstallation is done then it will lead to overwriting of the old chiscsi.rpmsave file.

b. Its advised to take a backup of chiscsi.conf file before you do an uninstallation and installation of new/same unified wire package. As re-installing/upgrading unified-wire package may lead to loss of chiscsi.conf file.

ii. Installation/uninstallation using source-tar-ball will neither remove the conf file nor rename it. It will always be intact. However it's recommended to always take a backup of your configuration file for both methods of installation.

A sample iSCSI configuration file will be installed in `/etc/chelsio-iscsi/chiscsi.conf`. This file should be edited using a standard text editor and customized to fit your environment.

2. Set iSCSI service to automatically start at bootup

The `chelsio-target` service scripts are installed to `/etc/init.d` and the parameters for the script are installed at `/etc/sysconfig/chiscsi`. The script is installed as a system service.

To auto-start the iSCSI target service at a certain runlevel, e.g. runlevel 3, `chkconfig` can be used on Red Hat and Novell / SuSE based systems as follows:

```
[root@host~]# chkconfig --level 3 chelsio-target on
```

The `chelsio-target` service scripts do basic checks before starting the iSCSI target service, loads the kernel module, and starts all the targets configured by default. It can also be used to stop the targets, and restart/reload configuration.

Note *For the script to execute properly, make sure the following flag is set on all kernel.org kernels.*

 # CONFIG_MODULE_FORCE_LOAD=y

3. Software/Driver Unloading

Use the following command to unload the module:

```
[root@host~]# rmmod chiscsi_t4
```

4. Software/Driver Configuration and Fine-tuning

The Chelsio iSCSI software needs configuration before it can become useful. The following sections describe how this is done.

There are two main components used in configuring the Chelsio iSCSI software: the **configuration file** and the **iSCSI control tool**. This section describes in some detail what they are and their relationship they have with one another.

4.1. Command Line Tools

There are two command line tools, one for control of the iSNS client and one for control of the iSCSI target nodes.

4.1.1. iscsictl

The Chelsio iSCSI control tool, `iscsictl`, is a Command Line Interface (CLI) user space program that allows administrators to:

- Start/Stop the iSCSI Target
- Start the iSNS client
- Get/Set the iSCSI driver global settings
- Get/Set/Remove the iSCSI Target configuration settings
- Retrieve active sessions' information of an iSCSI Target
- Manually flush data to the iSCSI Target disks
- Reload the iSCSI configuration file
- Update the iSCSI configuration file
- Save the current iSCSI configuration to a file

4.1.2. chisns

The Chelsio iSNS client, `chisns`, can be started independently of `iscsictl`.

4.2. iSCSI Configuration File

The iSCSI configuration file is the place where information about the Chelsio iSCSI software is stored. The information includes global data that pertains to all targets as well as information on each specific iSCSI target node. Most of the information that can be placed in the configuration file has default values that only get overwritten by the values set in the configuration file.There are only a few global configuration items that can be changed.

There are many specific parameters that can be configured, some of which are iSCSI specific and the rest being Chelsio specific. An example of an iSCSI specific item is "HeaderDigest" which is defaulted to "None" but can be overridden to "CRC32C". An example of a Chelsio specific configurable item is "ACL" (for Access Control List). "ACL" is one of the few items that have no default.

Before starting any iSCSI target, an iSCSI configuration file must be created. An easy way to create this file is to use the provided sample configuration file and modify it. This file can be named anything and placed in any directory but it must be explicitly specified when using iscsictl by using the -f option. To avoid this, put configuration file in the default directory (`/etc/chelsio-iscsi`) and name it the default file name (`chiscsi.conf`).

4.2.1. "On the fly" Configuration Changes

Parameters for the most part can be changed while an iSCSI node is running. However, there are exceptions and restrictions to this rule that are explained in a later section that describes the details of the iSCSI control tool `iscsictl`.

4.3. A Quick Start Guide for Target

This section describes how to get started quickly with a Chelsio iSCSI target. It includes:

- Basic editing of the iSCSI configuration file.
- Basic commands of the iSCSI control tool including how to start and stop a target.

4.3.1. A Sample iSCSI Configuration File

The default Chelsio iSCSI configuration file is located at `/etc/chelsio-iscsi/chiscsi.conf`. If this file doesn't already exist, then one needs to be created.

To configure an iSCSI target, there are three required parameters (in the form of key=value pairs) needed as follows:

- TargetName – A worldwide unique iSCSI target name.
- PortalGroup – The portal group tag associating with a list of target IP address (es) and port number(s) that service the login request. The format of this field is a Chelsio specific iSCSI driver parameter which is described in detail in the configuration file section.
- TargetDevice – A device served up by the associated target. A device can be:
    - A block device (for example, /dev/sda)
    - A virtual block device (for example, /dev/md0)
    - A RAM disk
    - A regular file

A target can serve multiple devices, each device will be assigned a Logical Unit Number (LUN) according to the order it is specified (i.e., the first device specified is assigned LUN 0, the second one LUN 1, …, and so on and so forth). Multiple TargetDevice key=value pairs are needed to indicate multiple devices.

Here is a sample of a minimum iSCSI target configuration located at `/etc/chelsio-iscsi/chiscsi.conf`:

```
target:
        TargetName=iqn.2006-02.com.chelsio.diskarray.san1
        TargetDevice=/dev/sda
        PortalGroup=1@192.0.2.178:3260
```

The `TargetDevice` value must match with the storage device in the system. The `PortalGroup` value must have a matching IP address of the Ethernet adapter card in the system.

For more information about TargetDevice configuration see **Target Storage Device Configuration**.

4.3.2. Basic iSCSI Control

Control of the Chelsio iSCSI software is done through `iscsictl`, the command line interface control tool. The following are the basic commands needed for effective control of the target.

Start Target: To start all of the iSCSI targets specified in the iSCSI configuration file, execute `iscsictl` with the "`-S`" option followed by "`target=ALL`".

```
[root@host~]# iscsictl -S target=ALL
```

To start a specific target execute `iscsictl` with "`-S`" followed by the target.

```
[root@host~]# iscsictl -S target=iqn.2006-02.com.chelsio.diskarray.san1
```

Stop Target: To stop the all the iSCSI target(s), execute `iscsictl` with "`-s`" option followed by "`target=ALL`".

```
[root@host~]# iscsictl -s target=ALL
```

To stop a specific target execute `iscsictl` with "`-s`" followed by the target name.

```
[root@host~]# iscsictl -s target=iqn.2006-02.com.chelsio.diskarray.san1
```

View Configuration: To see the configuration of all the active iSCSI targets, execute `iscsictl` with "`-c`" option.

```
[root@host~]# iscsictl -c
```

To see the more detailed configuration settings of a specific target, execute `iscsictl` with "`-c`" option followed by the target name.

```
[root@host~]# iscsictl -c target=iqn.2006-02.com.chelsio.diskarray.san1
```

View Global Settings: To see Chelsio global settings, execute `iscsictl` with "`-g`" option.

```
[root@host~]# iscsictl -g
```

Change Global Settings: To change Chelsio global settings, execute `iscsictl` with "`-G`" option.

```
[root@host~]# iscsictl -G iscsi_login_complete_time=300
```

View Help: To print help to stdout, execute `iscsictl` with "`-h`" option.

```
[root@host~]# iscsictl -h
```

4.4. The iSCSI Configuration File

The iSCSI configuration file consists of a series of blocks consisting of the following types of iSCSI entity blocks:

1. global
2. target

There can be only one global entity block whereas multiple target entity blocks are allowed. The global entity block is optional but there must be at least one target entity block.

An entity block begins with a block type (global or target). The content of each entity block is a list of parameters specified in a "key=value" format. An entity block ends at the beginning of the next entity block or at the end-of-file.

The parameter list in an entity block contains both:

- iSCSI parameters that override the default values
- Parameters that facilitate passing of control information to the iSCSI module

All lines in the configuration file that begin with “#” character are treated as comments and will be ignored. White space is not significant except in key=value pairs.

For the “key=value” parameters the <value> portion can be a single value or a list of multiple values. When <value> is a list of multiple values, they must be listed on one line with a comma “,” to separate their values. Another way to list the values instead of commas is to list their values as key=value pairs repeatedly, each on a new line, until they are all listed.

There are three categories of “key=value” parameter, the first category belongs to the global entity block whereas the second and third categories belong to target entity block:

1. The Chelsio Global Entity Settings of key=value pairs
2. The iSCSI Entity Settings of key=value pairs
3. The Chelsio Entity Settings of key=value pairs

The following sub-sections describe these three categories and list in tables the details of their key=value parameters.

4.4.1. Chelsio System Wide Global Entity Settings

Description

Chelsio System Wide Global Entity Parameters pass system control information to the iSCSI software which affects all targets in the same way. More detail of these parameters below can be found in a later section entitled “System Wide Parameters”.

Table of Chelsio Global Entity Settings

Key	Valid Values	Default Value	Multiple Values	Description
iscsi_auth_order	"ACL" "CHAP"	"CHAP"	No	Authorization order for login verification on the target. Valid only when a target's ACL_Enable=Yes **ACL**: ACL first then CHAP **CHAP**: CHAP first then ACL *Applies to Target(s) Only*
DISC_AuthMethod	"CHAP" "NONE"	None	No	To choose an authentication method for discovery phase.
DISC_Auth_CHAP_Policy	"Oneway" "Mutual"	"Oneway"	No	Oneway or Mutual (two-way) CHAP
DISC_Auth_CHAP_Target	"<user id>" :"<secret>"		Yes	CHAP user id and secret for the target. **<user id>** must be less than 256 characters. Commas "," are not allowed. **<secret>** must be between 6 and 255 characters. Commas "," are not allowed. The target user id and secret are used by the initiator to authenticate the target while doing mutual chap. *NOTE: The double quotes are required as part of the format.*
DISC_Auth_CHAP_Initiator	"<user id>" :"<secret>"		Yes	CHAP user id and secret for the initiator. **<user id>** must be less than 256 characters. Commas "," are not allowed. **<secret>** must be between 6 and 255 characters. Commas "," are not allowed. The initiator user id and secret are used by the target to authenticate the initiator. *NOTE: The double quotes are required as part of the format.*

iscsi_chelsio_ini_idstr	a string of maximum of 255 characters	"cxgb4i"	No	To enable additional optimization when Chelsio adapters and drivers are used at both ends (initiator and target) systems. Make sure the initiator name contain the substring set in iscsi_chelsio_ini_idstr when using Chelsio iscsi initiator driver.
iscsi_target_vendor_id	a string of maximum of 8 characters	"CHISCSI"	No	The target vendor ID part of the device identification sent by an iSCSI target in response of SCSI Inquiry command.
iscsi_login_complete_time	0 to 3600	300	No	Time allowed (in seconds) for the initiator to complete the login phase. Otherwise, the connection will be closed *NOTE: value zero means this check is NOT performed.*

4.4.2. iSCSI Entity Settings

Description

iSCSI Entity Parameters pass iSCSI protocol control information to the Chelsio iSCSI module. This information is unique for each entity block. The parameters follow the IETF iSCSI standard RFC 3720 in both definition and syntax. The descriptions below are mostly from this RFC.

Table of iSCSI Entity Settings

Key	Valid Values	Default Value	Multiple Values	Description
MaxConnections	1 to 65535	1	No	Initiator and target negotiate the maximum number of connections requested/acceptable.
InitialR2T	"Yes" "No"	"Yes"	No	To turn on or off the default use of R2T for unidirectional and the output part of bidirectional commands.
ImmediateData	"Yes" "No"	"Yes"	No	To turn on or off the immediate data.
FirstBurstLength	512 to 16777215 (2^{24} - 1)	65536	No	The maximum negotiated SCSI data in bytes of unsolicited data that an iSCSI initiator may send to a target during the execution of a single SCSI command.
MaxBurstLength	512 to 16777215 (2^{24} - 1)	262144	No	The maximum negotiated SCSI data in bytes, of a Data-In or a solicited Data-Out iSCSI sequence between the initiator and target.
DefaultTime2Wait	0 to 3600	2	No	The minimum time, in seconds, to wait before attempting an explicit / implicit logout or connection reset between initiator and target.

`DefaultTime2Retain`	0 to 3600	20	No	The maximum time, in seconds, after an initial wait.
`MaxOutstandingR2T`	1 to 65535	1	No	The maximum number of outstanding R2Ts per task.
`DataPDUInOrder`	`"Yes"` `"No"`	`"Yes"`	No	To indicate the data PDUs with sequence must be at continuously increasing order or can be in any order. *Chelsio only supports "Yes".*
`DataSequenceInOrder`	`"Yes"` `"No"`	`"Yes"`	No	To indicate the Data PDU sequences must be transferred in continuously non-decreasing sequence offsets or can be transferred in any order. *Chelsio only supports "Yes".*
`ErrorRecoveryLevel`	0 to 2	0	No	To negotiate the recovery level supported by the node. *Chelsio only supports 0.*
`HeaderDigest`	`"None"` `"CRC32C"`	`"None"`	Yes	To enable or disable iSCSI header Cyclic integrity checksums.
`DataDigest`	`"None"` `"CRC32C"`	`"None"`	Yes	To enable or disable iSCSI data Cyclic integrity checksums.
`AuthMethod`	`"CHAP"` and `"None"`	`"None, CHAP"`	Yes	To choose an authentication method during login phase.
`TargetName`	`"<target name>"`		No	A worldwide unique iSCSI target name. *Target only.*
`TargetAlias`	`"<target alias>"`		No	A human-readable name or description of a target. It is not used as an identifier, nor is it for authentication. *Target only.*
`MaxRecvDataSegmentLength`	512 to 16777215 (2^{24} - 1)	8192	No	To declare the maximum data segment length in bytes it can receive in an iSCSI PDU.
`OFMarker`	`"Yes"` `"No"`	`"No"`	No	To turn on or off the initiator to target markers on the connection. *Chelsio only supports "No".*
`IFMarker`	`"Yes"` `"No"`	`"No"`	No	To turn on or off the target to initiator markers on the connection. *Chelsio only supports "No".*
`OFMarkInt`	1 to 65535	2048	No	To set the interval for the initiator to target markers on a connection.
`IFMarkInt`	1 to 65535	2048	No	To set the interval for the target to initiator markers on a connection.

4.4.3. Chelsio Entity Settings

Description

Chelsio Entity Parameters pass control information to the Chelsio iSCSI module. The parameters are specific to Chelsio'simplementation of the iSCSI node (target or initiator) and are unique for each entity block. The parameters consist of information that can be put into three categories:

1. Challenge Handshake Authentication Protocol (CHAP).
2. Target specific settings. All of the following parameters can have multiple instances in one target entity block (i.e., they can be declared multiple times for one particular target).

- Portal Group
- Storage Device

  Access Control List (ACL)

Table of Chelsio Entity Settings

Key	Valid Values	Default Value	Multiple Values	Description
		Chelsio CHAP Parameter (Target)		
`Auth_CHAP_Target`	`"<user id>":"<secret>"`		No	CHAP user id and secret for the target. **<user id>** must be less than 256 characters. Commas "," are not allowed. **<secret>** must be between 6 and 255 characters. Commas "," are not allowed. The target user id and secret are used by the initiator to authenticate the target while doing mutual chap. *NOTE: The double quotes are required as part of the format.*
`Auth_CHAP_Initiator`	`"<user id>":"<secret>"`		Yes	CHAP user id and secret for the initiator. **<user id>** must be less than 256 characters. Commas "," are not allowed. **<secret>** must be between 6 and 255 characters. Commas "," are not allowed. The initiator user id and secret are used by the target to authenticate the initiator. *NOTE: The double quotes are required as part of the format.*
`Auth_CHAP_ChallengeLength`	16 to 1024	16	No	CHAP challenge length
`Auth_CHAP_Policy`	`"Oneway" or "Mutual"`	`"Oneway"`	No	Oneway or Mutual (two-way) CHAP
		Chelsio Target Specific Parameter		

`PortalGroup`	`<portal group tag>@<target IP address>[:<port number>]`
	.
	.
	.
	`[,<target IP`
			Yes	The portal group name associates the given target with the given list of IP addresses (and optionally, port numbers) for servicing login requests. It"s required to have at least one per target.
				
				<portal group tag> is a unique tag identifying the portal group. It must be a positive integer.

	address> [:<port number>]] [,timeout= <timeout value in milliseconds>] [,[portalgrou ptag1, portalgroupta g2,… portalgroupta gn]			**<target IP address>** is the IP address associated with the portal group tag. **<port number>** is the port number associated with the portal group tag. It is optional and if not specified the well-known iSCSI port number of 3260 is used. **<timeout>** is optional, it applies to all the portals in the group. The timeout value is in milliseconds and needs to be multiple of 100ms. It is used to detect loss of communications at the iSCSI level. *NOTE: There can be multiple target IP address/port numbers per portal group tag. This enables a target to operate on multiple interfaces for instance.* **<portalgrouptagX>**The portalgroup to which login requests should be redirected to. *NOTE: There can be multiple redirection target portalgroups specified for a particular target portal group and the redirection will happen to these in a round robin manner.*
ShadowMode	"Yes" "No"	"No"	No	To turn ShadowMode on or off for iSCSI Target Redirection
TargetSessionMaxCmd	1 to 2048	64	No	The maximum number of outstanding iSCSI commands per session.
TargetDevice*	<path/name> [,FILE|MEM|BL K] [,NULLRW] [,SYNC] [,RO] [,size=xMB] [,ID=xxxxxx] [,WWN=xxxxxxx xx] [,SN= xxxxxx]		No	A device served up by the associated target. The device mode can be a: • Block Device (e.g. /dev/sda) • Virtual Block Device (e.g. /dev/md0) • RamDisk • Regular File **<path/name>** is the path to the device - with the exception of when a RAM Disk is specified, where it is a unique name given to the device. If multiple RAM Disks are used for a target then each name must be unique within the target. **NULLRW** specifies that random data is returned for reads, and for writes data is dropped. Useful for testing network performance. **SYNC** specifies that the device will function in the write-through mode (i.e.,

				the data will be flushed to the device before the response is returned to the initiator). *NOTE: SYNC is only applicable with FILE mode.* **RO** specifies the device as a read-only device. **FILE** specifies this device should be accessed via the kernel‟s VFS layer. This mode is the most versatile, and it is the default mode in the cases where there is no mode specified. **BLK** specifies this device should be accessed via the kernel‟s block layer. This mode is suitable for high-speed storage device such as RAID Controllers. **MEM** specifies this device should be created as a RAM Disk. **size**=xMB is used with "MEM", to specify the RamDisk size. If not specified, the default RamDisk size is 16MB (16 Megabytes). The minimum value of x is 1 (1MB) and the maximum value is limited by system memory. **SN** is a 16 character unique value. **ID** is a 24 character unique value. **WWN** is a 16 character unique value. It is recommended when using a multipath aware initiator , the optional ID (short form for SCSI ID), SN and WWN values should be set manually for the TargetDevice. These values will be returned in Inquiry response (VPD 0x83). Multiple TargetDevice key=value pairs are needed to indicate multiple devices. There can be multiple devices for any particular target. Each device will be assigned a Logical Unit Number (LUN) according to the order it is specified (i.e., the first device specified is assigned LUN 0, the second one LUN 1, …, and so on and so forth). *NOTE: **FILE** mode is the most versatile mode, if in doubt use **FILE** mode.*
`ACL_Enable`	`"Yes"` `"No"`	`"No"`	No	Defines if Chelsio'sAccess Control List (ACL) method will be enforced on the target:

				Yes: ACL is enforced on the target **No:** ACL is not enforced on the target *NOTE*: ACL flag is not allowed to be updated on the fly. Target must be restarted for new ACL flag to take effect.
ACL	[iname=<name1>][;<sip=<sip1>][;dip=<dip1>][;lun=<lun_list:permissions>]		Yes	The ACL specifies which initiators and how they are allowed to access the LUNs on the target. **iname=<Initiator Name>** specifies one or more initiator names, the name must be a fully qualified iSCSI initiator name. **sip=<Source IP address>** specifies one or more IP addresses the initiators are connecting from. **Dip=<Destination IP address>** specifies one or more IP addresses that the iSCSI target is listening on (i.e., the target portal IP addresses). *NOTE: when configuring an ACL at least one of the above three must be provided:* • *iname, and/or* • *sip, and/or* • *dip.* **lun=<lun list>:<permission>** controls how the initiators access the luns. The supported value for **<lun list>** is **ALL**. **<permissions>** can be: **R**: Read Only **RW** or **WR**: Read and Write If permissions are specified then the associated LUN list is required. If no **lun=<lun list>:[R|RW]** is specified then it defaults to **ALL:RW**. *NOTE: For the Chelsio Target Software release with lun-masking included,* ***<lun list>** is in the format of **<0..N | 0~N | ALL>*** *Where:* ***0..N**: only one value from **0** through **N*** ***0~N**: a range of values between **0** through **N*** ***ALL**: all currently supported LUNs.* *Multiple lists of LUN numbers are allowed. When specifying the list separate the LUN ranges by a comma.*
RegisteriSNS	"Yes" "No"	"Yes"	No	To turn on or off exporting of target information via iSNS

4.4.4. Sample iSCSI Configuration File

Following is a sample configuration file. While using iSCSI node (target), irrelevant entity block can be removed or commented.

```
#
# Chelsio iSCSI Global Settings
#
global:
        iscsi_login_complete_time=300
        iscsi_auth_order=CHAP
        DISC_AuthMethod=None
        DISC_Auth_CHAP_Policy=Oneway
        DISC_Auth_CHAP_Target="target_id1":"target_secret1"
        DISC_Auth_CHAP_Initiator="initiator_id1":"initiator_sec1"
#
# an iSCSI Target "iqn.2006-02.com.chelsio.diskarray.san1"
# being served by the portal group "5". Setup as a RAM Disk.
#
target:
        TargetName=iqn.2006-02.com.chelsio.diskarray.san1
        # lun 0: a ramdisk with default size of 16MB
        TargetDevice=ramdisk,MEM
        PortalGroup=5@192.0.2.178:3260
#
# an iSCSI Target "iqn.2005-8.com.chelsio:diskarrays.san.328"
# being served by the portal group "1" and "2"
#
target:
        #
        # iSCSI configuration
        #
        TargetName=iqn.2005-8.com.chelsio:diskarrays.san.328
        TargetAlias=iTarget1
        MaxOutstandingR2T=1
        MaxRecvDataSegmentLength=8192
        HeaderDigest=None,CRC32C
        DataDigest=None,CRC32C
```

```
ImmediateData=Yes
InitialR2T=No
FirstBurstLength=65535
MaxBurstLength=262144

#
# Local block devices being served up
# lun 0 is pointed to /dev/sda
# lun 1 is pointed to /dev/sdb

TargetDevice=/dev/sda,ID=aabbccddeeffgghh,WWN=aaabbbcccdddeeef
TargetDevice=/dev/sdb

#
# Portal groups served this target
#

PortalGroup=1@102.50.50.25:3260
PortalGroup=2@102.60.60.25:3260

#
# CHAP configuration
#
Auth_CHAP_Policy=Mutual
Auth_CHAP_target="iTarget1ID":"iTarget1Secret"

Auth_CHAP_Initiator="iInitiator1":"InitSecret1"
Auth_CHAP_Initiator="iInitiator2":"InitSecret2"
Auth_CHAP_ChallengeLength=16

#
# ACL configuration
#
# initiator "iqn.2006-02.com.chelsio.san1" is allowed full access
# to this target
ACL=iname=iqn.2006-02.com.chelsio.san1

# any initiator from IP address 102.50.50.101 is allowed full access
# of this target
ACL=sip=102.50.50.101
```

```
        # any initiator connected via the target portal 102.60.60.25 is
        # allowed full access to this target
        ACL=dip=102.60.60.25

        # initiator "iqn.2005-09.com.chelsio.san2" from 102.50.50.22 and
        # connected via the target portal 102.50.50.25 is allowed read only
        # access of this target
        ACL=iname=iqn.2006-
02.com.chelsio.san2;sip=102.50.50.22;dip=102.50.50.25;lun=ALL:R
```

4.5. Challenge-Handshake Authenticate Protocol (CHAP)

CHAP is a protocol that is used to authenticate the peer of a connection and uses the notion of a challenge and response, (i.e., the peer is challenged to prove its identity).

The Chelsio iSCSI software supports two CHAP methods: **one-way** and **mutual**. CHAP is supported for both login and discovery sessions.

4.5.1. Normal Session CHAP Authentication

For a normal Session, the CHAP authentication is configured on a per-target basis.

4.5.2. Oneway CHAP authentication

With **one-way** CHAP (also called unidirectional CHAP) the target uses CHAP to authenticate the initiator. The initiator does not authenticate the target. This method is the default method.

For **one-way** CHAP, the initiator CHAP id and secret are configured and stored on a per-initiator with Chelsio Entity parameter "Auth_CHAP_Initiator".

4.5.3. Mutual CHAP authentication

With **mutual** CHAP (also called bidirectional CHAP), the target and initiator use CHAP to authenticate each other.

For **mutual** CHAP, in addition to the initiator CHAP id and secret, the target CHAP id and secret are required. They are configured and stored on a per target basis with Chelsio Entity parameter "Auth_CHAP_Target".

4.5.4. Adding CHAP User ID and Secret

A single `Auth_CHAP_Target` key and multiple `Auth_CHAP_Initiator` keys could be configured per target:

```
target:
      TargetName=iqn.2006-02.com.chelsio.diskarray.san1
      TargetDevice=/dev/sda PortalGroup=1@192.0.2.178:8000
      Auth_CHAP_Policy=Oneway
      Auth_CHAP_Initiator="remoteuser1":"remoteuser1_secret"
      Auth_CHAP_Initiator="remoteuser2":"remoteuser2_secret"
      Auth_CHAP_Target="targetid1":"target1_secret"
```

In the above example, target `iqn.2005-com.chelsio.diskarray.san1` has been configured to authenticate two initiators, and its own id and secret are configured for use in the case of mutual CHAP.

4.5.5. AuthMethod and Auth_CHAP_Policy Keys

By setting the iSCSI keys `AuthMethod` and `Auth_CHAP_Policy`, a user can choose whether to enforce CHAP and if mutual CHAP needs to be performed.

The `AuthMethod` key controls if an initiator needs to be authenticated or not. The default setting of `AuthMethod` is `None,CHAP`

The `Auth_CHAP_Policy` key controls which CHAP authentication (one-way or mutual) needs to be performed if CHAP is used. The default setting of `Auth_CHAP_Policy` is `Oneway`

On an iSCSI node, with `AuthMethod=None,CHAP`

- `Auth_CHAP_Policy=Oneway`, Chelsio iSCSI target will accept a relevant initiator if it does
  a) no CHAP
  b) CHAP Oneway or
  c) CHAP Mutual

- `Auth_CHAP_Policy=Mutual`, the Chelsio iSCSI target will accept a relevant initiator if it does
  a) no CHAP or
  b) CHAP Mutual

With `AuthMethod=None`, regardless the setting of the key Auth_CHAP_Policy, the Chelsio iSCSI target will only accept a relevant initiator if it does no CHAP.

With `AuthMethod=CHAP`, CHAP is enforced on the target:

i. `Auth_CHAP_Policy=Oneway`, the iSCSI target will accept a relevant initiator only if it does
  a) CHAP Oneway or
  b) CHAP Mutual

ii. `Auth_CHAP_Policy=Mutual`, the iSCSI node will accept a relevant initiator only if it does
  a) CHAP Mutual

4.5.6. Discovery Session CHAP

CHAP authentication is also supported for the discovery sessions where an initiator queries of all available targets.

Discovery session CHAP is configured through the global section in the configuration file. List of keys to provision discovery CHAP are:

- DISC_AuthMethod: None or CHAP.
- DISC_Auth_CHAP_Policy: Oneway or Mutual (i.e., two-way) authentication.
- DISC_Auth_CHAP_Target: target CHAP user id and secret
- DISC_Auth_CHAP_Initiator: initiator CHAP user id and secret.

Sample:

```
#
# Chelsio iSCSI Global Settings
#
global:
        DISC_AuthMethod=CHAP
        DISC_Auth_CHAP_Policy=Mutual
        DISC_Auth_CHAP_Target="target_id1":"target_secret1"
        DISC_Auth_CHAP_Initiator="initiator_id1":"initiator_sec1"
```

4.6. Target Access Control List (ACL) Configuration

The Chelsio iSCSI target supports iSCSI initiator authorization via an Access Control List (ACL).

ACL configuration is supported on a per-target basis. The creation of an ACL for a target establishes:

- Which iSCSI initiators are allowed to access it
- The type of the access: read-write or read-only
- Possible SCSI layer associations of LUNs with the initiator

More than one initiator can be allowed to access a target and each initiator‟s access rights can be independently configured.

The format for ACL rule is as follows:

```
ACL=[iname=<initiator name>][;<sip=<source ip addresses>]
    [;dip=<destination ip addresses>][;lun=<lun_list>:<permissions>]

 target:
       TargetName=iqn.2006-02.com.chelsio.diskarray.san1
       TargetDevice=/dev/sda

       PortalGroup=1@102.50.50.25:3260
       PortalGroup=2@102.60.60.25:3260

       # initiator "iqn.2006-02.com.chelsio.san1" is allowed
       # full read-write access to this target
       ACL=iname=iqn.2006-02.com.chelsio.san1

       # any initiator from IP address 102.50.50.101 is allowed full
       # read-write access of this target
       ACL=sip=102.50.50.101

       # any initiator connected via the target portal 102.60.60.25
       # is allowed full read-write access to this target
       ACL=dip=102.60.60.25

       # initiator "iqn.2005-09.com.chelsio.san2" from 102.50.50.22
       # and connected via the target portal 102.50.50.25 is allowed
       # read only access of this target
       ACL=iname=iqn.2006-
       02.com.chelsio.san2;sip=102.50.50.22;dip=102.50.50.25;lun=ALL:R
```

4.6.1. ACL Enforcement

To toggle ACL enforcement on a per-target base, a Chelsio keyword `ACL_Enable` is provided:

- Setting `ACL_Enable=Yes` enables the target to perform initiator authorization checking for all the initiators during login phase. And in addition, once the initiator has been authorized to access the target, the access rights will be checked for each individual LU the initiator trying to access.
- Setting `ACL_Enable=No` disable the target to perform initiator authorization checking.

When a target device is marked as read-only (RO), it takes precedence over ACL's write permission (i.e., all of ACL write permission of an initiator is ignored).

4.7. Target Storage Device Configuration

An iSCSI Target can support one or more storage devices. The storage device can either be the built-in RAM disk or actual backend storage.

Configuration of the storage is done through the Chelsio configuration file via the key-value pair `TargetDevice`.

When option `NULLRW` is specified, on writes the data is dropped without being copied to the storage device, and on reads the data is not actually read from the storage device but instead random data is used. This option is useful for measuring network performance.

The details of the parameters for the key `TargetDevice` are found in the table of Chelsio Entity Settings section earlier in this document.

4.7.1. RAM Disk Details

For the built-in RAM disk:

- The minimum size of the RAM disk is 1 Megabyte (MB) and the maximum is limited by system memory.
- To use a RAM disk with a Windows Initiator, it is recommended to set the size >= 16MB.

To configure an ramdisk specify `MEM` as the device mode:

```
TargetDevice=<name>,MEM,size=xMB
```

Where:	`<name>`	Is a unique name given to the RAM disk. This name identifies this particular ramdisk. If multiple RAM disks are configured for the same target, the name must be unique for each RAM Disk.
	`x`	Is the size of the RAM disk in MB. It's an integer between (1-max), where max is limited by system memory. If this value is not specified the default value is 16 MB.

```
target:
#<snip>
  # 16 Megabytes RAM Disk named ramdisk1
  TargetDevice=ramdisk1,MEM,size=16MB
#<snip>
```

4.7.2. FILE Mode Storage Device Details

The FILE mode storage device is the most common and versatile mode to access the actual storage attached to the target system:

- The FILE mode can accommodate both block devices and virtual block devices.
- The device is accessed in the exclusive mode. The device should not be accessed (or active) in any way on the target system.
- Each device should be used for one and only one iSCSI target.
- “SYNC” can be used with FILE mode to make sure the data is flushed to the storage device before the Target responds back to the Initiator.

To configure a FILE storage device specify `FILE` as the device mode:

```
TargetDevice=<path to the storage device>[,FILE][,SYNC]
```

Where:	`<path>`	Is the path to the actual storage device, such as /dev/sdb for a block device or /dev/md0 for a software RAID. The path must exist in the system.
	`SYNC`	When specified, the Target will flush all the data in the system cache to the storage driver before sending response back to the Initiator.

4.7.3. Example Configuration of FILE Mode Storage

Below is an example:

```
target:
#<snip>
 # software raid /dev/md0 is accessed in FILE mode
TargetDevice=/dev/md0,FILE
#<snip>
```

4.7.4. BLK Mode Storage Device Details

The BLK mode storage device is suitable for high-speed storage attached to the target system:

- The BLK mode can accommodate only block devices.
- The device is accessed in the exclusive mode. The device should not be accessed (or active) in any way on the target system.
- Each device should be used for one and only one iSCSI target.

To configure a block storage device specify BLK as the device mode:

```
TargetDevice=<path to the storage device>,BLK
```

Where:	`<path>`	Is the path to the actual storage device, such as /dev/sdb. The path must exist in the system.

```
target:
#<snip>
 # /dev/sdb is accessed in BLK mode
 TargetDevice=/dev/sdb,BLK
#<snip>
```

4.7.5. Multi-path Support

To enable multi-path support on the initiator, it is highly recommended that the following options are specified:

- *[,ID=xxxxxx]*: SCSI ID, a twenty-four (24) bytes alpha-numeric string
- *[,WWN=xxxxxxxxx]*: SCSI World Wide Name (WWN), a sixteen (16) bytes alpha-numeric string
- *[,SN= xxxxxx]*: SCSI SN, a sixteen (15) bytes alpha-numeric string.

The user should make sure the three values listed above are the same for the target LUNs involved in the multipath.

4.8. Target Redirection Support

An iSCSI Target can redirect an initiator to use a different IP address and port (often called a portal) instead of the current one to connect to the target. The redirected target portal can either be on the same machine, or a different one.

4.8.1. ShadowMode for Local vs. Remote Redirection

The *ShadowMode* setting specifies whether the Redirected portal groups should be present on the same machine or not. If *ShadowMode* is enabled, the redirected portal groups are on a different system. If it is disabled then the redirected portal groups must be present on the same system otherwise the target would fail to start.

Below is an example with *ShadowMode* enabled:

```
target:
 #<snip>
     # any login requests received on 10.193.184.81:3260 will be
     # redirected to 10.193.184.85:3261.

     PortalGroup=1@10.193.184.81:3260,[2]
     PortalGroup=2@10.193.184.85:3261

     # the PortalGroup "2" is NOT presented on the same system.
```

```
     ShadowMode=Yes
 #<snip>
```

Below is an example with *ShadowMode* disabled:

```
target:
 #<snip>
     # any login requests received on 10.193.184.81:3260 will be
     # redirected to 10.193.184.85:3261

    PortalGroup=1@10.193.184.81:3260,[2]
    PortalGroup=2@10.193.184.85:3261
    # the PortalGroup "2" IS present on the same system
    ShadowMode=No
#<snip>
```

4.8.2. Redirecting to Multiple Portal Groups

The Chelsio iSCSI Target Redirection allows redirecting all login requests received on a particular portal group to multiple portal groups in a round robin manner.

Below is an example Redirection to Multiple Portal Groups:

```
target:
 #<snip>
     # any login requests received on 10.193.184.81:3260 will be
     # redirected to 10.193.184.85:3261 and 10.193.184.85:3262 in a
     # Round Robin Manner.

     PortalGroup=1@10.193.184.81:3260,[2,3]
     PortalGroup=2@10.193.184.85:3261
     PortalGroup=3@10.193.184.85:3262
     ShadowMode=No
#<snip>
```

4.9. The command line interface tools "iscsictl" & "chisns"

4.9.1. iscsictl

`iscsictl` is the tool Chelsio provides for controlling the iSCSI target. It is a Command Line Interface (CLI) that is invoked from the console. Its usage is as follows:

```
iscsictl <options> <mandatory parameters> [optional parameters]
```

The mandatory and optional parameters are the **key=value** pair(s) defined in RFC3720, or the **var=const** pair(s) defined for Chelsio iSCSI driver implementation. In this document, the key=value is referred to as "pair", and var=const is referred to as "parameter" to clarify between iSCSI protocol‟s pair value(s), and Chelsio iSCSI driver‟s parameter value(s). Note that all **value** and **const** are case sensitive.

4.9.2. chisns

`chisns` is the command line tool for controlling the iSNS client. This is a simple tool that starts the iSNS client with a client and server parameter.

4.9.3. iscsictl options

Options	Mandatory Parameters	Optional Parameters	Description
`-h`			Display the help messages.
`-v`			Display the version.
`-f`	`<[path/] filename>`		Specifies a pre-written iSCSI configuration text file, used to start, update, save, or reload the iSCSI node(s). This option must be specified with one of the following other options: "**-S**" or "**-W**". For the "**-S**" option "**-f**" must be specified first. All other options will ignore this "**-f**" option. If the "**-f**" option is not specified with the commands above the default configuration file will be used. It‟s name and location is: `/etc/chelsio-iscsi/chiscsi.conf` The configuration file path and filename must conform to Linux standards. For the format of the iSCSI configuration file, please see "Format of The iSCSI Configuration File" section earlier in this document.
`-k`	`<key>[=<val>]`		Specifies an iSCSI Entity or Chelsio Entity parameter. This option can be specified after "**-c**" option to retrieve a parameter setting..

`-c`	`target=<name>`
	`[,name2`
	`.`
	`.`
	`.`
	`,<nameN>]`
			Display the Chelsio iSCSI target configuration.
			
			target=<name> parameter:
			Where **name** is the name of the node whose information will be returned. **name** can be one or more string of names, separated by a comma,
			<name1[,name2,…,nameN] | ALL>
			
			A **name** of **ALL** returns information on all targets. **ALL** is a reserved string that must be uppercase.
			
			Example:
			iscsictl -c target=iqn.com.cc.it1
			Iscsictl -c target=iqn.com.cc.target1 -k

			TargetAlias The **<name>** parameter can also be specified as one or more parameter on the same command line, separated by a comma, **target=<name1>, <name2>, … ,<nameN>** The **target=<name>** parameter(s) are optional and if not specified all active Chelsio iSCSI targets(s) configuration(s) will be displayed. If **target=ALL** is specified or no parameters are specified the output will be abbreviated. Specify specific targets to get detailed configuration data. If the **target=<name>** option is specified, the -k <key> option can optionally be specified along with this option to display only the selected entity parameter setting. Example: **iscsictl -c target=iqn.com.cc.target1 -k HeaderDigest**
`-F`		`target=<name>` `-k lun=<value>`	Flush the cached data to the target disk(s). **target=<name>** parameter: Where **name** is the name of the target to be flushed. **name** can be one or more string of names, separated by a comma, **<name1[,name2,…,nameN] | ALL>** A **name** of **ALL** will cause all the target data to be flushed. **ALL** is a reserved string that must be uppercase. The **target=name** parameter is optional. If no **target=name** parameter is specified, it is the same as specifying **target=ALL**. The -k lun=<value> option is optional. It can be used to further specify a particular lun to be flushed. Example: To flush all the targets in the system: **iscsictl -F** To flush a particular target: **iscsictl -F target=iqn.com.cc.it1** To flush only the lun 0 of a particular target: **iscsictl -F target=iqn.com.cc.it1 -k lun=0**
`-g`			Display the Chelsio iSCSI Global Entity Settings.
`-G`	`<var=const>`		Set the Chelsio iSCSI Global Entity Settings. **var=const** parameter: Where **var=const** can be any key listed under Chelsio Global Entity Settings. Example: **iscsictl -G iscsi_auth_order=ACL**

			The **var=const** parameter(s) are mandatory. If the **var=const** parameter is not specified, the command will be denied. If any of the specified **var=const** parameter is invalid, the command will reject only the invalid parameters, but will continue on and complete all other valid parameters if any others are specified.
`-s`	`target=<name>`		Stop the specified active iSCSI targets. **target=<name>** parameter: *See the description of option -c for the target=<name> parameter definition.* The **target=<name>** parameter is mandatory. If no **target=<name>** parameter is specified, the command will be denied. If the **target=<name>** parameter is specified, only the specified targets from the **target=<name>** parameters will be stopped. If **target=ALL** is specified, all active targets will stop.
`-S`	`target=<name>`		Start or reload the iSCSI targets. **target=<name>** parameter: Where **name** is the name of the target(s) that will be started or reloaded. The **target=<name>** parameter can be specified as one or more parameter on the same command line, separated by a space, **target=<name1> target=<name2> … target=<nameN>** The **target=<name>** parameter can also be, **target=ALL** A **name** of **ALL** starts or reloads all targets specified in the configuration file. **ALL** is a reserved string that must be uppercase. The **target=<name>** parameter is optional. If this command line option is specified without the -f option, the default configuration file `/etc/chelsio-iscsi/chiscsi.conf` will be used. Rules, 1. If the **target=<name>** parameter is specified, only the targets from the list will be started or reloaded. 2. If **target=ALL** is specified, all targets specified from the iSCSI configuration file will be started or reloaded. 3. If the **target=<name>** parameter is not specified, all active targets configurations will be reloaded from the configuration file while those targets are running. All non-active targets specified will not be loaded / started.

			For Rules 1-3, if the specified targets are currently active (running), they will get reloaded. For Rules 1 & 2, if the specified targets are not currently active, they will be started. For Rules 2 & 3, please note the differences – they are not the same! The global settings are also reloaded from the configuration file with this option.
`-r`	`target=<name>`	`-k initiator=<name>`	Retrieve active iSCSI sessions under a target. **target=<name>** parameter: Where **name** must be a single target name. If **target=<name>** parameter is specified as target=<name>, the sessions can be further filtered based on the remote node name with optional –k initiator=<name> option. Examples: **iscsictl -r target=iqn.com.cc.it1** **iscsictl -r target=iqn.com.cc.it1 -k initiator=iqn.com.cc.ii1** The first **target=<name>** parameter is mandatory. If it is not specified, the command will be denied.
`-D`	`<Session handle in hex>`		Drop initiator session. This option should be specified with the handle of the session (in hex) that needs to be dropped. The session handle can be retrieved using the previous mentioned iscsictl option (-r used to retrieve active iSCSI sessions under a target).
`-W`			Overwrite the specified iSCSI configuration file with ONLY the current iSCSI global settings and the active iSCSI targets" configuration to the specified iSCSI configuration file. *Will delete any non-active targets' configuration from the specified file.* The -f option MUST be specified along with this option.
`-h`			Display the help messages.
	`server=<IP address>[:<port>]`	`id=<isns entity id> query=<query interval>`	Start the Chelsio iSNS client. **server=<IP address>[:<port>]** where **server** is the iSNS server address. The port is optional and if it"s not specified it defaults to 3205. The server with the ip address is mandatory and if it"s not specified the, the command will be denied. **id=<isns entity id>** where **id** is the iSNS entity ID used to register with the server. It defaults to <hostname>. **query=<query interval>** where **query** is the initiator query interval (in seconds). It defaults to 60 seconds. Examples: chisns server=192.0.2.10

			chisns server=192.0.2.10:3205 id=isnscln2 query=30 In the first example the minimum command set is given where the IP address of the iSNS server is specified. In the second example a fully qualified command is specified by also setting three optional parameters. Here, the mandatory IP address and the corresponding optional port number are specified. Also set is the iSNS entity ID to "isnscln2" as well as the query interval to 30 seconds.

4.9.4. chisns options

Options	Mandatory Parameters	Optional Parameters	Description
-h			Display the help messages.
	server=<IP address>[:<port>]	id=<isns entity id> query=<query interval>	Start the Chelsio iSNS client. **server=<IP address>[:<port>]** where **server** is the iSNS server address. The port is optional and if it's not specified it defaults to 3205. The server with the ip address is mandatory and if it's not specified the, the command will be denied. **id=<isns entity id>** where **id** is the iSNS entity ID used to register with the server. It defaults to <hostname>. **query=<query interval>** where **query** is the initiator query interval (in seconds). It defaults to 60 seconds. Examples: chisns server=192.0.2.10 chisns server=192.0.2.10:3205 id=isnscln2 query=30 In the first example the minimum command set is given where the IP address of the iSNS server is specified. In the second example a fully qualified command is specified by also setting three optional parameters. Here, the mandatory IP address and the corresponding optional port number are specified. Also set is the iSNS entity ID to 'isnscln2' as well as the query interval to 30 seconds.

4.10. Rules of Target Reload (i.e. "on the fly" changes)

After a target has been started its settings can be modified via reloading of the configuration file (i.e., `iscsictl -S`).

The following parameters cannot be changed once the target is up and running otherwise the target reload would fail:

- TargetName
- TargetSessionMaxCmd
- ACL_Enable
- ACL

The following parameters **CAN** be changed by reloading of the configuration file. The new value will become effective **IMMEDIATELY** for all connections and sessions:

- TargetDevice
- PortalGroup

The following parameter **CAN** be changed by reloading of the configuration file. The new value will **NOT** affect any connections and sessions that already completed login phase:

- TargetAlias
- MaxConnections
- InitialR2T
- ImmediateData
- FirstBurstLength
- MaxBurstLength
- MaxOutstandingR2T
- HeaderDigest
- DataDigest
- MaxRecvDataSegmentLength
- AuthMethod
- Auth_CHAP_Initiator
- Auth_CHAP_Target
- Auth_CHAP_ChallengeLength
- Auth_CHAP_Policy

The following parameters **SHOULD NOT** be changed because only one valid value is supported:

- DataPDUInOrder (support only "Yes")
- DataSequenceInOrder (support only "Yes")
- ErrorRecoveryLevel (support only "0")
- OFMarker (support only "No")
- IFMarker (support only "No")

The following parameters can be changed but would not have any effect because they are either not supported or they are irrelevant:

- DefaultTime2Wait (not supported)
- DefaultTime2Retain (not supported)

- OFMarkInt (irrelevant because OFMarker=No)
- IFMarkInt (irrelevant because IFMarker=No)

4.11. System Wide Parameters

The Chelsio Global Entity Settings are system wide parameters that can be controlled through the configuration file or the use of the command line "`iscsictl -G`". The finer points of some of these parameters are described in detail here:

4.11.1. iscsi_login_complete_time

Options: An integer value between 0 and 3600 (seconds). Default value is 300 (seconds).

This is the login timeout check. The parameter controls the maximum time (in seconds) allowed to the initiator to complete the login phase. If a connection has been in the login phase longer than the set value, the target will drop the connection.

Value zero turns off this login timeout check.

4.11.2. iscsi_auth_order

Options: "ACL" or "CHAP", defaults to "CHAP"

On an iSCSI target when `ACL_Enable` is set to `Yes`, `iscsi_auth_order` decides whether to perform CHAP first then ACL or perform ACL then CHAP.

- **ACL:** When setting `iscsi_auth_order=ACL`, initiator authorization will be performed at the start of the login phase for an iSCSI normal session: upon receiving the first iscsi_login_request, the target will check its ACL. If this iSCSI connection does not match any ACL provisioned, the login attempt will be terminated.

- **CHAP**: When setting `iscsi_auth_order=CHAP`, initiator authorization will be performed at the end of the login phase for an iSCSI normal session: before going to the full feature phase, the target will check its ACL. If this iSCSI connection does not match any ACL provisioned, the login attempt will be terminated.

Note *iscsi_auth_order has no meaning when* `ACL_Enable` *is set to* `No` *on a target.*

4.11.3. iscsi_target_vendor_id

Options: A string of maximum of 8 characters, defaults to `CHISCSI`

The `iscsi_target_vendor_id` is part of the device identification sent by an iSCSI target in response of a SCSI Inquiry request.

4.11.4. iscsi_chelsio_ini_idstr

Options: A string of maximum of 255 characters, defaults to “cxgb4i”.

For an iscsi connection, more optimization can be done when both initiator and target are running Chelsio adapters and drivers.

This string is used to verify the initiator name received, and identify if the initiator is running Chelsio drivers: if the initiator name contains the same substring as `iscsi_chelsio_ini_idstr` it is assumed the initiator is running with the Chelsio iscsi initiator driver and additional offload optimization is performed.

4.12. Performance Tuning

i. **See performance tuning section** in the Unified Wire chapter for generic performance settings.
ii. Next, load the iSCSI PDU offload target driver (*chiscsi_t4*) and run the *chiscsi_set_affinity.sh* script to map iSCSI worker threads to different CPUs.

```
[root@host~]# chiscsi_set_affinity.sh
```

X. iSCSI PDU Offload Initiator

1. Introduction

The Chelsio Unified Wire series of adapters support iSCSI acceleration and iSCSI Direct Data Placement (DDP) where the hardware handles the expensive byte touching operations, such as CRC computation and verification, and direct DMA to the final host memory destination:

- **iSCSI PDU digest generation and verification**
  On transmit -side, Chelsio hardware computes and inserts the Header and Data digest into the PDUs. On receive-side, Chelsio hardware computes and verifies the Header and Data digest of the PDUs.

- **Direct Data Placement (DDP)**
  Chelsio hardware can directly place the iSCSI Data-In or Data-Out PDU's payload into pre-posted final destination host-memory buffers based on the Initiator Task Tag (ITT) in Data-In or Target Task Tag (TTT) in Data-Out PDUs.

- **PDU Transmit and Recovery**
  On transmit-side, Chelsio hardware accepts the complete PDU (header + data) from the host driver, computes and inserts the digests, decomposes the PDU into multiple TCP segments if necessary, and transmit all the TCP segments onto the wire. It handles TCP retransmission if needed.
  On receive-side, Chelsio hardware recovers the iSCSI PDU by reassembling TCP segments, separating the header and data, calculating and verifying the digests, then forwarding the header to the host. The payload data, if possible, will be directly placed into the pre-posted host DDP buffer. Otherwise, the data will be sent to the host too.

The *cxgb4i* driver interfaces with open-iSCSI initiator and provides the iSCSI acceleration through Chelsio hardware wherever applicable.

1.1. Hardware Requirements

1.1.1. Supported adapters

The following are the currently shipping Chelsio adapters that are compatible with iSCSI PDU Offload Initiator Software:

- T62100-LP-CR
- T6225-CR
- T520-BT
- T580-CR
- T520-LL-CR
- T520-CR
- T580-LP-CR

- T540-CR
- T420-CR
- T440-CR
- T422-CR
- T404-BT
- T420-BCH
- T440-LP-CR
- T420-BT
- T420-LL-CR
- T420-CX

1.2. Software Requirements

1.2.1. Linux Requirements

Currently the iSCSI PDU Offload Initiator software is available for the following versions:

- RHEL 7.3, 3.10.0-514.el7
- RHEL 7.3, 3.10.0-514.el7.ppc64le (POWER8 LE)
- RHEL 7.3, 4.5.0-15.el7.aarch64 (ARM64)
- RHEL 7.2, 3.10.0-327.el7
- RHEL 6.8, 2.6.32-642.el6
- RHEL 6.7, 2.6.32-573.el6
- SLES 12 SP2, 4.4.21-69-default
- SLES 12 SP1, 3.12.49-11-default
- SLES 11 SP4, 3.0.101-63-default
- Ubuntu 16.04.1, 4.4.0-47-generic
- Ubuntu 14.04.4, 4.2.0-27-generic
- Kernel.org linux-4.8

Other kernel versions have not been tested and are not guaranteed to work.

2. Software/Driver Loading

Important *Please ensure that all inbox drivers are unloaded before proceeding with unified wire drivers.*

The driver must be loaded by the root user. Any attempt to load the driver as a regular user will fail.

Run the following command to load the driver:

```
[root@host~]# modprobe cxgb4i
```

If loading of cxgb4i displays "unkown symbols found" error in dmesg, follow the steps mentioned below:

i. Kill iSCSI daemon *iscsid*
ii. View all the loaded iSCSI modules

```
[root@host~]# lsmod | grep iscsi
```

iii. Now, unload them using the following command:

```
[root@host~]# rmmod <modulename>
```

iv. Finally reload the *cxgb4i* driver.

3. Software/Driver Unloading

To unload the driver, execute the following commands:

```
[root@host~]# rmmod cxgb4i
[root@host~]# rmmod libcxgbi
```

4. Software/Driver Configuration and Fine-tuning

4.1. Accelerating open-iSCSI Initiator

The following steps need to be taken to accelerate the open-iSCSI initiator:

4.1.1. Configuring interface (*iface)* file

Create an interface file located under *iface* directory for the new transport class *cxgb4i* in the following format:

```
iface.iscsi_ifacename = <iface file name>
iface.hwaddress = <MAC address>
iface.transport_name = cxgb4i
iface.net_ifacename = <ethX>
iface.ipaddress = <iscsi ip address>
```

E.g.:-

```
iface.iscsi_ifacename = cxgb4i.00:07:43:04:5b:da
iface.hwaddress = 00:07:43:04:5b:da
iface.transport_name = cxgb4i
iface.net_ifacename = eth3
iface.ipaddress = 102.2.2.137
```

Alternatively, you can create the file automatically by executing the following command:

```
[root@host~]# iscsiadm -m iface
```

Here,

- `iface.iscsi_ifacename` denotes the name of interface file in `/etc/iscsi/ifaces/`.
- `iface.hwaddress` denotes the MAC address of the Chelsio interface via which iSCSI traffic will be running.
- `iface.transport_name` denotes the transport name, which is cxgb4i.
- `iface.net_ifacename` denotes the Chelsio interface via which iSCSI traffic will be running.
- `iface.ipaddress` denotes the IP address which is assigned to the interface.

Note

i. The interface file needs to be created in `/etc/iscsi/iscsid.conf`*.*

ii. If `iface.ipaddress` *is specified, it needs to be either the same as the ethX's IP address or an address on the same subnet. Make sure the IP address is unique in the network.*

4.1.2. Discovery and Login

i. Starting iSCSI Daemon

Start Daemon from */sbin* by using the following command:

```
[root@host~]# iscsid -f -d 3
```

Note *If* `iscsid` *is already running, then kill the service and start it as shown above after installing the Chelsio Unified Wire package.*

ii. Discovering iSCSI Targets

To discovery an iSCSI target execute a command in the following format:

```
iscsiadm -m discovery -t st -p <target ip address>:<target port no> -I
<cxgb4i iface file name>
```

E.g.:-

```
[root@host~]# iscsiadm -m discovery -t st -p 102.2.2.155:3260 -I
cxgb4i.00:07:43:04:5b:da
```

iii. Logging into an iSCSI Target

Log into an iSCSI target using the following format:

```
iscsiadm -m node -T <iqn name of target> -p <target ip address>:<target port
no> -I  <cxgb4i iface file name> -l
```

E.g.:-

```
[root@host~]# iscsiadm -m node -T iqn.2004-05.com.chelsio.target1 -p
102.2.2.155:3260,1 -I cxgb4i.00:07:43:04:5b:da -l
```

If the login fails with an error message in the format of `ERR! MaxRecvSegmentLength <X> too big. Need to be <= <Y>.` in dmesg, edit the `iscsi/iscsid.conf` file and change the setting for `MaxRecvDataSegmentLength`:

```
node.conn[0].iscsi.MaxRecvDataSegmentLength = 8192
```

Important *Always take a backup of `iscsid.conf` file before installing Chelsio Unified Wire Package. Although the file is saved to `iscsid.rpmsave` after uninstalling the package using RPM, you are still advised to take a backup.*

iv. Logging out from an iSCSI Target

Log out from an iSCSI Target by executing a command in the following format:

```
iscsiadm -m node -T <iqn name of target> -p <target ip address>:<target port
no> -I  <cxgb4i iface file name> -u
```

E.g.:-

```
[root@host~]# iscsiadm -m node -T iqn.2004-05.com.chelsio.target1 -p
102.2.2.155:3260,1 -I cxgb4i.00:07:43:04:5b:da -u
```

Note *Other options can be found by typing* `iscsiadm --help`

4.2. Auto login from cxgb4i initiator at OS bootup

For iSCSI auto login (via *cxgb4i*) to work on OS startup, please add the following line to `start()` in `/etc/rc.d/init.d/iscsid` file on RHEL:

```
modprobe -q cxgb4i
```

E.g.:-

```
force_start() {
    echo -n $"Starting $prog: "
    modprobe -q iscsi_tcpmodprobe -q ib_iser
    modprobe -q cxgb4i
    modprobe -q cxgb3i
    modprobe -q bnx2i
    modprobe -q be2iscsi
    daemon brcm_iscsiuio
    daemon $prog
    retval=$?
    echo
    [ $retval -eq 0 ] && touch $lockfile
    return $retval
}
```

XI. Crypto Offload

1. Introduction

Chelsio's Terminator 6 (T6) Unified Wire ASIC enables concurrent secure communication and secure storage with support for integrated TLS/SSL/DTLS and inline cryptographic functions, leveraging the proprietary TCP/IP offload engine. Chelsio's full offload TLS/SSL/DTLS is uniquely capable of 100Gb line-rate performance. In addition, the accelerator can be used in a traditional co-processor Lookaside mode to accelerate TLS/SSL, IPsec, SMB 3.X crypto, data at rest encryption/decryption, and data-deduplication fingerprint computation.

1.1. Hardware Requirements

1.1.1. Supported adapters

The following are the currently shipping Chelsio adapters that are compatible with Chelsio's Crypto Offload driver:

- T62100-LP-CR
- T6225-CR

1.2. Software Requirements

1.2.1. Linux Requirements

Currently Chelsio's Crypto Offload driver is available for the following versions:

- Kernel.org linux-4.8

Other kernel versions have not been tested and are not guaranteed to work.

2. Software/Driver Loading

Important *Ensure that the kernel is installed with Crypto components (refer* Pre-requisites section*) before proceeding with Unified Wire installation.*

To load the Crypto Offload driver in Co-processor mode (*chcr*), run the following command:

```
[root@host~]# modprobe chcr
```

To load the Crypto Offload driver in Inline mode, load the Network driver in TOE mode:

```
[root@host~]# modprobe t4_tom
```

3. Software/Driver Configuration and Fine-tuning

3.1. Supported Algorithms (Co-processor)

To view the complete list of supported cryptographic algorithms use the following command:

```
[root@host~]# cat /proc/crypto|grep -i chcr
```

3.1. TLS Port Configuration (Inline)

In order to use Chelsio crypto offload in Inline mode, use the appropriate openssl.cnf file:

```
[root@host~]# cd /usr/chssl/openssl/
[root@host~]# cp openssl-tls.cnf openssl.cnf
```

A total of up to 12 TCP ports can be configured for TLS offload,each port id corresponding to a specific security protocol. Following are the supported TCP ports and their corresponding security protocols:

Port ID	Security Protocol
443	HTTPS
989	FTPS protocol (data)
990	FTPS protocol (control)
992	Telnet protocol
993	Internet Message Access protocol (IMAPS)
994	Internet Relay Chat (IRCS)
995	Post Office Protocol 3 (POP3S)
1364	IBM COnnect: Direct FTP+
4116	Smartcard-TLS
5349	STUN protocol for NAT traversal
6514	Syslog over TLS
8531	Windows Server Update Services

To set security protocol(s), follow the steps mentioned below:

i. Configure MSS to 16k

```
[root@host~]# echo 16384 > /proc/sys/toe/toeX_tom/mss
```

ii. Define the respective TCP port(s) using the *tls_ports* configuration interface.

```
[root@host~]# echo "<port_id> <port_id> <port_id>...<port_id>" >
/proc/sys/toe/toeX_tom/tls_ports
```

E.g.:

```
[root@beagle3 ~]# echo 16384 > /proc/sys/toe/toe1_tom/mss
[root@beagle3 ~]# cat /proc/sys/toe/toe1_tom/mss
16384
[root@beagle3 ~]# echo "443 989 990 992 993 994 995" > /proc/sys/toe/toe1_tom/tls_ports
[root@beagle3 ~]# cat /proc/sys/toe/toe1_tom/tls_ports
443     989     990     992     993     994     995     0       0       0       0       0
```

3.2. Chelsio OpenSSL (Co-processor, Inline)

OpenSSL which supports Chelsio crypto offload is installed as part of Unified Wire package. It is installed in */usr/chssl/bin*

- To determine the hash value of a file:

```
[root@host~]# cd /usr/chssl/bin
[root@host~]# ./openssl dgst -engine af_alg -sha1 <file>
```

E.g.:

```
[root@beagle1 ~]# cd /usr/chssl/bin/
[root@beagle1 bin]# ./openssl dgst -engine af_alg -sha1 /root/RandomChars.txt
engine "af_alg" set.
SHA1(/root/RandomChars.txt)= c82a2203f5ed76a5ac0365ad3913e4c99d45d898
```

- To start TLS offload Server

```
[root@host~]# cd /usr/chssl/bin
[root@host~]# ./openssl s_server -key <key_file> -cert <cert_file> -accept
443 -cipher AES128-GCM-SHA256  -WWW
```

E.g.:

```
[root@HEATHER ~]# ./openssl s_server -key /root/server.key -cert /root/server.crt
-accept 443 -cipher AES128-GCM-SHA256  -WWW
```

- To start TLS offload Client:

```
[root@host~]# cd /usr/chssl/bin
[root@host~]# ./openssl s_time -connect <tls_server_ip>:<tls_server_port>  -
www /<file>
```

E.g.:

```
[root@beagle3 bin]# ./openssl s_time -connect 102.1.1.154:443  -www /1G
No CIPHER specified
Collecting connection statistics for 30 seconds
**********************************************

47 connections in 12.42s; 3.78 connections/user sec, bytes read 50465867843
47 connections in 31 real seconds, 1073741869 bytes read per connection


Now timing with session id reuse.
starting
rrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrrr

47 connections in 12.32s; 3.81 connections/user sec, bytes read 50465867843
47 connections in 31 real seconds, 1073741869 bytes read per connection
```

4. Software/Driver Unloading

To unload Crypto Offload driver in Co-processor mode, run the following command:

```
[root@host~]# rmmod chcr
```

To unload Crypto Offload driver in Inline mode, unload the network driver in TOE mode. See **Software/Driver Unloading** section in **Network (NIC/TOE)** chapter for more information.

XII. Data Center Bridging (DCB)

1. Introduction

Data Center Bridging (DCB) refers to a set of bridge specification standards, aimed to create a converged Ethernet network infrastructure shared by all storage, data networking and traffic management services. An improvement to the existing specification, DCB uses priority-based flow control to provide hardware-based bandwidth allocation and enhances transport reliability.

One of DCB's many benefits includes low operational cost, due to consolidated storage, server and networking resources, reduced heat and noise, and less power consumption. Administration is simplified since the specifications enable transport of storage and networking traffic over a single unified Ethernet network.

Note *In this release, ETS bandwidth management will not work when Unified Wire traffic is run with IEEE DCBx enabled.*

1.1. Hardware Requirements

1.1.1. Supported adapters

The following are the currently shipping Chelsio adapters that are compatible with Chelsio's DCB feature:

- T62100-LP-CR
- T6225-CR
- T520-LL-CR
- T520-CR
- T580-CR
- T580-LP-CR
- T420-CR
- T420-LL-CR

1.2. Software Requirements

1.2.1. Linux Requirements

Currently Chelsio's DCB feature is available for the following versions:

- RHEL 7.3, 3.10.0-514.el7
- RHEL 7.2, 3.10.0-327.el7
- RHEL 6.8, 2.6.32-642.el6
- RHEL 6.7, 2.6.32-573.el6
- SLES 12 SP2, 4.4.21-69-default
- SLES 12 SP1, 3.12.49-11-default
- SLES 11 SP4, 3.0.101-63-default

- Ubuntu 16.04.1, 4.4.0-47-generic
- Ubuntu 14.04.4, 4.2.0-27-generic
- Kernel.org linux-4.8

Other kernel versions have not been tested and are not guaranteed to work.

2. Software/Driver Loading

Important *Please ensure that all inbox drivers are unloaded before proceeding with unified wire drivers.*

Before proceeding, please ensure that Unified Wire Installer is installed with DCB support as mentioned in **CLI mode (individual drivers)** section of Unified Wire Installer chapter.

Network (*cxgb4; t4_tom* for full offload support) and FCoE Initiator (*csiostor*) drivers must be loaded in order to enable DCB feature. Also, the drivers must be loaded by the root user. Any attempt to load the drivers as a regular user will fail. Run the following commands:

```
[root@host~]# modprobe cxgb4
[root@host~]# modprobe t4_tom
[root@host~]# modprobe csiostor
```

Once the storage and networking traffic are started simultaneously, they will honor DCB settings defined on the switch.

3. Software/Driver Unloading

To disable DCB feature, unload FCoE Initiator and Network drivers:

```
[root@host~]# rmmod csiostor
[root@host~]# rmmod cxgb4
```

Note *If t4_tom is loaded, please reboot machine after unloading FCoE Initiator and Network drivers.*

4. Software/Driver Configuration and Fine-tuning

4.1. Configuring Cisco Nexus 5010 switch

4.1.1. Configuring the DCB parameters

Note *By default the Cisco Nexus switch enables DCB functionality and configures PFC for FCoE traffic making it no drop with bandwidth of 50% assigned to FCoE class of traffic and another 50% for the rest (like NIC). If you wish to configure custom bandwidth, then follow the procedure below.*

In this procedure, you may need to adjust some of the parameters to suit your environment, such as VLAN IDs, Ethernet interfaces, and virtual Fibre Channel interfaces.

To enable PFC, ETS, and DCB functions on a Cisco Nexus 5010 series switch:

i. Open a terminal configuration setting.

```
switch# config terminal
switch(config)#
```

ii. Configure qos class-maps and set the traffic priorities: NIC uses priority 0 and FcoE uses priority 3.

```
switch(config)#class-map type qos class-nic
switch(config-cmap-qos)# match cos 0
switch(config-cmap-qos)# class-map type qos class-fcoe
switch(config-cmap-qos)# match cos 3
```

iii. Configure queuing class-maps.

```
switch(config)#class-map type queuing class-nic
switch(config-cmap-que)#match qos-group 2
```

iv. Configure network-qos class-maps.

```
switch(config)#class-map type network-qos class-nic
switch(config-cmap-nq)#match qos-group 2
```

v. Configure qos policy-maps.

```
switch(config)#policy-map type qos policy-test
switch(config-pmap-qos)#class type qos class-nic
switch(config-pmap-c-qos)#set qos-group 2
```

vi. Configure queuing policy-maps and assign network bandwidth. Divide the network bandwidth between FcoE and NIC traffic.

```
switch(config)#policy-map type queuing policy-test
switch(config-pmap-que)#class type queuing class-nic
switch(config-pmap-c-que)#bandwidth percent 50
switch(config-pmap-c-que)#class type queuing class-fcoe
switch(config-pmap-c-que)#bandwidth percent 50
switch(config-pmap-c-que)#class type queuing class-default
switch(config-pmap-c-que)#bandwidth percent 0
```

vii. Configure network-qos policy maps and set up the PFC for no-drop traffic class.

```
switch(config)#policy-map type network-qos policy-test
switch (config-pmap-nq)#class type network-qos class-nic
switch(config-pmap-nq-c)#pause no-drop
```

Note *By default FCoE is set to pause no drop. In such a trade off, one may want to set NIC to drop instead.*

viii. Apply the new policy (PFC on NIC and FcoE traffic) to the entire system.

```
switch(config)#system qos
switch(config-sys-qos)#service-policy type qos input policy-test
switch(config-sys-qos)#service-policy type queuing output policy-test
switch(config-sys-qos)#service-policy type queuing input policy-test
switch(config-sys-qos)#service-policy type network-qos policy-test
```

4.1.2. Configuring the FCoE/FC ports

In this procedure, you may need to adjust some of the parameters to suit your environment, such as VLAN IDs, Ethernet interfaces, and virtual Fibre Channel interfaces

i. Following steps will enable FCoE services on a particular VLAN and does a VSAN-VLAN mapping. Need not do these steps every time, unless a new mapping has to be created.

```
switch(config)# vlan 2
switch(config-vlan)# fcoe vsan 2
switch(config-vlan)#exit
```

ii. Following steps help in creating a virtual fibre channel (VFC) and binds that VFC to a Ethernet interface so that the Ethernet port begins functioning as a FCoE port.

```
switch(config)# interface vfc 13
switch(config-if)#bind interface ethernet 1/13
switch(config-if)# no shutdown
switch(config-if)#exit
switch(config)#vsan database
switch(config-vsan-db)# vsan 2
switch(config-vsan-db)#vsan 2 interface vfc 13
switch(config-vsan-db)#exit
```

Note *If you are binding the VFC to a MAC address instead of an ethernet port then make sure the ethernet port is part of both default VLAN and FCoE VLAN.*

iii. Assign VLAN ID to the Ethernet port on which FCoE service was enabled in step1.

```
switch(config)# interface ethernet 1/13
switch(config-if)#switchport mode trunk
switch(config-if)#switchport trunk allowed vlan 2
switch(config-if)#no shutdown
switch(config)#exit
```

iv. Enabling DCB:

```
switch(config)# interface ethernet 1/13
switch(config-if)# priority-flow-control mode auto
switch(config-if)# flowcontrol send off
switch(config-if)# flowcontrol receive off
switch(config-if)# lldp transmit
switch(config-if)# lldp receive
switch(config-if)# no shutdown
```

v. On the FC Ports, if a FC target is connected then perform the following steps -

```
switch(config)#vsan database
switch(config-vsan-db)#vsan 2
switch(config-vsan-db)# vsan 2 interface fc 2/2
switch(config-vsan-db)#exit
switch(config)interface fc 2/2

switch(config-if)# switchport mode auto
switch(config-if)# switchport speed auto
switch(config-if)# no shutdown.
```

vi. If you have not created a zone then make sure the default-zone permits the VSAN created, otherwise the initiator and the target on that particular VSAN although FLOGI'd into the switch will not talk to each other. To enable it, execute the below command.

```
switch(config)# zone default-zone permit vsan 2
```

4.2. Configuring the Brocade 8000 switch

i. Configure LLDP for FCoE.Example of configuring LLDP for 10-Gigabit Ethernet interface.

```
switch(config)#protocol lldp
switch(conf-lldp)#advertise dcbx-fcoe-app-tlv
switch(conf-lldp)#advertise dcbx-fcoe-logical-link-tlv
```

ii. Create a CEE Map to carry LAN and SAN traffic if it does not exist. Example of creating a CEE map.

```
switch(config)# cee-map default
switch(conf-cee-map)#priority-group-table 1 weight 40 pfc
switch(conf-cee-map)#priority-group-table 2 weight 60
switch(conf-cee-map)#priority-table 2 2 2 1 2 2 2 2
```

iii. Configure the CEE interface as a Layer 2 switch port. Example of configuring the switch port as a 10-Gigabit Ethernet interface.

```
switch(config)#interface tengigabitethernet 0/16
switch(config-if-te-0/16)#switchport
switch(config-if-te-0/16)#no shutdown
switch(config-if)#exit
```

iv. Create an FCoE VLAN and add an interface to it. Example of creating a FCoE VLAN and adding a single interface.

```
switch(config)#vlan classifier rule 1 proto fcoe encap ethv2
switch(config)#vlan classifier rule 2 proto fip encap ethv2
switch(config)#vlan classifier group 1 add rule 1
switch(config)#vlan classifier group 1 add rule 2
switch(config)#interface vlan 1002
switch(conf-if-vl-1002 )#fcf forward
switch(conf-if-vl-1002 )#interface tengigabitethernet 0/16
switch(config-if-te-0/16)#switchport
switch(config-if-te-0/16)#switchport mode converged
switch(config-if-te-0/16)#switchport converged allowed vlan add 1002
switch(config-if-te-0/16)#vlan classifier activate group 1 vlan 1002
switch(config-if-te-0/16)#cee default
switch(config-if-te-0/16)#no shutdown
switch(config-if-te-0/16)#exit
```

Note *Unlike cisco, only one VLAN ID can carry FCoE traffic for now on Brocade 8000. It is their limitation.*

v. Save the Configuration

```
switch#copy running-config startup-config
```

5. Running NIC & iSCSI Traffic together with DCBx

Note *Please refer* iSCSI PDU Offload Initiator *chapter to configure iSCSI Initiator.*

Use the following procedure to run NIC and iSCSI traffic together with DCBx enabled.

i. Identify the VLAN priority configured for NIC and iSCSI class of traffic on the switch.
ii. Create VLAN interfaces for running NIC and iSCSI traffic, and configure corresponding VLAN priority.

Example:

Switch is configured with a VLAN priority of 2 and 5 for NIC and iSCSI class of traffic respectively. NIC traffic is run on VLAN10 and iSCSI traffic is run on VLAN20.

Assign proper VLAN priorities on the interface (here eth5), using the following commands on the host machine:

```
[root@host~]# vconfig set_egress_map eth5.10 0 2
[root@host~]# vconfig set_egress_map eth5.20 5 5
```

XIII. FCoE Full Offload Initiator

1. Introduction

Fibre Channel over Ethernet (FCoE) is a mapping of Fibre Channel over selected full duplex IEEE 802.3 networks. The goal is to provide I/O consolidation over Ethernet, reducing network complexity in the Datacenter. Chelsio FCoE initiator maps Fibre Channel directly over Ethernet while being independent of the Ethernet forwarding scheme. The FCoE protocol specification replaces the FC0 and FC1 layers of the Fibre Channel stack with Ethernet. By retaining the native Fibre Channel constructs, FCoE will integrate with existing Fibre Channel networks and management software.

1.1. Hardware Requirements

1.1.1. Supported adapters

The following are the currently shipping Chelsio adapters that are compatible with FCoE full offload Initiator driver:

- T520-BT
- T520-LL-CR
- T520-CR

1.2. Software Requirements

1.2.1. Linux Requirements

Currently the FCoE full offload Initiator driver is available for the following versions:

- RHEL 7.3, 3.10.0-514.el7
- RHEL 7.2, 3.10.0-327.el7
- RHEL 6.8, 2.6.32-642.el6
- RHEL 6.7, 2.6.32-573.el6
- SLES 12 SP2, 4.4.21-69-default
- SLES 12 SP1, 3.12.49-11-default
- SLES 11 SP4, 3.0.101-63-default
- Ubuntu 16.04.1, 4.4.0-47-generic
- Ubuntu 14.04.4, 4.2.0-27-generic
- Kernel.org linux-4.8

Other kernel versions have not been tested and are not guaranteed to work.

2. Software/Driver Loading

Important *Please ensure that all inbox drivers are unloaded before proceeding with unified wire drivers.*

The driver must be loaded by the root user. Any attempt to load the driver as a regular user will fail.

To load the driver, execute the following:

```
[root@host~]# modprobe csiostor
```

3. Software/Driver Unloading

To unload the driver:

```
[root@host~]# modprobe -r  csiostor
```

Note *If multipath services are running, unload of FCoE driver is not possible. Stop the multipath service and then unload the driver.*

4. Software/Driver Configuration and Fine-tuning

4.1. Configuring Cisco Nexus 5010 and Brocade switch

To configure various Cisco and Brocade switch settings, please refer **Software/Driver Configuration and Fine-tuning** section of **Data Center Bridging (DCB)** chapter.

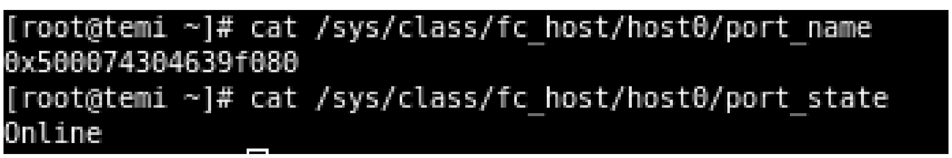
4.2. FCoE fabric discovery verification

4.2.1. Verifying Local Ports

Once connected to the switch, use the following command to see if the FIP has gone through and a VN_Port MAC address has been assigned.

Verify if all the FCoE ports are online/ready and a successful FIP has taken place using the following command. The **wwpn** and **state** of the initiator local port can be found under sysfs.

```
[root@host~]# cat /sys/class/fc_host/hostX/port_name
```



Note

- *The hosts under fc_host depends on the number of ports on the adapter used.*
- *Inorder to identify chelsio fc_host from other vendor fc_host, the WWPN always begins with **0x5000743***

Alternatively, the local port information can also be found using:

```
[root@host~]# cxgbtool stor -a <adapter_no> --show-lnode
```

```
[root@temi ~]# cxgbtool stor -a /dev/csiostor0 --show-lnode
*****************[Index:  0]****************
LNode Device ID: 2950
VNPI     : 0xb86
FCFI     : 0xba3
MAC      : 0E-FC-01-67-00-0D

Port Id : 0
Nport id: 67000d
State    : READY

WWPN     : 50:00:74:30:46:39:f0:80
WWNN     : 50:00:74:30:46:39:f0:00
NPIV     : SUPPORTED
Total VPorts :  0

No.of RNodes : 10

Common Service Params:
        Rcv size: 2068
        ED-TOV  : 2000

Class Service Params:
Class 1: NOT SUPPORTED
Class 2: NOT SUPPORTED
Class 3: SUPPORTED
        Initiator ctl           : 0
        Recipient ctl           : 0
        Rcv size                : 2068
        Total concurrent seq    : 0
        EE Credit               : 0
        Open Sequence per Exchange: 0

Class 4: NOT SUPPORTED
*****************[Index:  1]****************
LNode Device ID: 68484
VNPI     : 0xb84
FCFI     : 0xba1
MAC      : 0E-FC-01-67-00-02

Port Id : 1
Nport id: 670002
State    : READY

WWPN     : 50:00:74:30:46:3a:71:80
WWNN     : 50:00:74:30:46:3a:71:00
NPIV     : SUPPORTED
Total VPorts :  0

No.of RNodes : 10
```

4.2.2. Verifying the target discovery

To view the list of targets discovered on a particular FCoE port, use the following commands:

i. Check for the adapter number using the following command.

```
[root@host~]# cxgbtool stor -s
```

ii. To check the list of targets discovered on a particular FCoE port, first determine the WWPN of the initiator local port under *sysfs*. The hosts under *fc_host* depends on the number of ports on the adapter used.

```
[root@host~]# cat /sys/class/fc_host/hostX/port_name
```

iii. After finding the localport, go to the corresponding Remote port under sysfs **# cat /sys/class/fc_remote_ports/rport-X:B:R** where X is the Host ID, B is the bus ID and R is the remote port.

```
[root@temi ~]# cat /sys/class/fc_remote_ports/rport-0\:0-0/roles
Fabric Port
[root@temi ~]# cat /sys/class/fc_remote_ports/rport-0\:0-1/roles
Directory Server
[root@temi ~]# cat /sys/class/fc_remote_ports/rport-0\:0-2/roles
Management Server
[root@temi ~]# cat /sys/class/fc_remote_ports/rport-0\:0-3/roles
FCP Initiator
[root@temi ~]# cat /sys/class/fc_remote_ports/rport-0\:0-4/roles
FCP Initiator
[root@temi ~]# cat /sys/class/fc_remote_ports/rport-0\:0-9/roles
FCP Target
[root@temi ~]#
```

Note *R can correspond to NameServer, Management Server and other initiator ports logged in to the switch and targets.*

Alternatively, the local ports can also be found using *cxgbtool:*

```
[root@host~]# cxgbtool stor -a <adapter no>   --show-lnode
```

After finding out the WWPN of the local node, to verify the list of discovered targets, use the following command.

```
[root@host~]# cxgbtool stor -a <adapter_no> --show-rnode --wwn=<wwpn of
lnode>
```

```
[root@temi ~]# cxgbtool stor -a /dev/csiostor0 --show-rnode --wwn=50:00:74:30:46:3b:73:80

*****************[Index:   0]***************
SSNI            : 0x780
VNPI            : 0xb84
FCFI            : 0xba2
WWPN            : 20:06:00:05:73:d5:7a:ff
WWNN            : 20:05:00:05:73:d5:7a:c1
Nport id        : FFFFFE
State           : READY
FCP Flags       : 0
Role            : Fabric

Class Service Params:
Class 1: NOT SUPPORTED
Class 2: NOT SUPPORTED
Class 3: SUPPORTED
Class 4: NOT SUPPORTED

*****************[Index:   1]***************
SSNI            : 0x781
VNPI            : 0xb84
FCFI            : 0xba2
WWPN            : 25:0d:00:05:73:d5:7a:c0
WWNN            : 20:05:00:05:73:d5:7a:c1
Nport id        : FFFFFC
State           : READY
FCP Flags       : 0
Role            : Name-Server

Class Service Params:
Class 1: NOT SUPPORTED
Class 2: NOT SUPPORTED
Class 3: SUPPORTED
Class 4: NOT SUPPORTED

*****************[Index:   2]***************
SSNI            : 0x783
VNPI            : 0xb84
FCFI            : 0xba2
WWPN            : 25:0b:00:05:73:d5:7a:c0
WWNN            : 20:05:00:05:73:d5:7a:c1
Nport id        : FFFFFA
State           : READY
FCP Flags       : 0
Role            : N-Port

Class Service Params:
Class 1: NOT SUPPORTED
Class 2: NOT SUPPORTED
Class 3: SUPPORTED
```

4.3. Formatting the LUNs and Mounting the Filesystem

Use *lsscsi –g* to list the LUNs discovered by the initiator

```
[root@host~]# lsscsi -g
```

```
[root@temi ~]# lsscsi -g
[0:0:0:0]    disk    NETAPP   LUN              8010  /dev/sda   /dev/sg0
[0:0:0:1]    disk    NETAPP   LUN              8010  /dev/sdb   /dev/sg1
[0:0:0:2]    disk    NETAPP   LUN              8010  /dev/sdc   /dev/sg2
[0:0:0:3]    disk    NETAPP   LUN              8010  /dev/sdd   /dev/sg3
[0:0:0:4]    disk    NETAPP   LUN              8010  /dev/sde   /dev/sg4
[0:0:0:5]    disk    NETAPP   LUN              8010  /dev/sdf   /dev/sg5
[0:0:0:6]    disk    NETAPP   LUN              8010  /dev/sdg   /dev/sg6
[0:0:0:7]    disk    NETAPP   LUN              8010  /dev/sdh   /dev/sg7
[0:0:0:8]    disk    NETAPP   LUN              8010  /dev/sdi   /dev/sg8
[0:0:0:9]    disk    NETAPP   LUN              8010  /dev/sdj   /dev/sg9
[0:0:0:10]   disk    NETAPP   LUN              8010  /dev/sdk   /dev/sg10
[0:0:0:11]   disk    NETAPP   LUN              8010  /dev/sdl   /dev/sg11
[0:0:0:12]   disk    NETAPP   LUN              8010  /dev/sdm   /dev/sg12
[0:0:0:13]   disk    NETAPP   LUN              8010  /dev/sdn   /dev/sg13
[0:0:0:14]   disk    NETAPP   LUN              8010  /dev/sdo   /dev/sg14
[0:0:0:15]   disk    NETAPP   LUN              8010  /dev/sdp   /dev/sg15
[0:0:0:16]   disk    NETAPP   LUN              8010  /dev/sdq   /dev/sg16
[0:0:0:17]   disk    NETAPP   LUN              8010  /dev/sdr   /dev/sg17
[0:0:0:18]   disk    NETAPP   LUN              8010  /dev/sds   /dev/sg18
[0:0:0:19]   disk    NETAPP   LUN              8010  /dev/sdt   /dev/sg19
[1:0:0:0]    disk    NETAPP   LUN              8010  /dev/sdu   /dev/sg20
[1:0:0:1]    disk    NETAPP   LUN              8010  /dev/sdv   /dev/sg21
[1:0:0:2]    disk    NETAPP   LUN              8010  /dev/sdw   /dev/sg22
[1:0:0:3]    disk    NETAPP   LUN              8010  /dev/sdx   /dev/sg23
[1:0:0:4]    disk    NETAPP   LUN              8010  /dev/sdy   /dev/sg24
[1:0:0:5]    disk    NETAPP   LUN              8010  /dev/sdz   /dev/sg25
[1:0:0:6]    disk    NETAPP   LUN              8010  /dev/sdaa  /dev/sg26
[1:0:0:7]    disk    NETAPP   LUN              8010  /dev/sdab  /dev/sg27
[1:0:0:9]    disk    NETAPP   LUN              8010  /dev/sdac  /dev/sg28
[1:0:0:10]   disk    NETAPP   LUN              8010  /dev/sdad  /dev/sg29
[1:0:0:11]   disk    NETAPP   LUN              8010  /dev/sdae  /dev/sg30
[1:0:0:12]   disk    NETAPP   LUN              8010  /dev/sdaf  /dev/sg31
[1:0:0:15]   disk    NETAPP   LUN              8010  /dev/sdag  /dev/sg32
[3:0:0:0]    disk    NETAPP   LUN              8010  /dev/sdah  /dev/sg33
[3:0:0:1]    disk    NETAPP   LUN              8010  /dev/sdai  /dev/sg34
[3:0:0:2]    disk    NETAPP   LUN              8010  /dev/sdaj  /dev/sg35
[3:0:0:3]    disk    NETAPP   LUN              8010  /dev/sdak  /dev/sg36
[3:0:0:4]    disk    NETAPP   LUN              8010  /dev/sdal  /dev/sg37
[3:0:0:5]    disk    NETAPP   LUN              8010  /dev/sdam  /dev/sg38
[3:0:0:6]    disk    NETAPP   LUN              8010  /dev/sdan  /dev/sg39
[3:0:0:7]    disk    NETAPP   LUN              8010  /dev/sdao  /dev/sg40
[3:0:0:8]    disk    NETAPP   LUN              8010  /dev/sdap  /dev/sg41
[3:0:0:9]    disk    NETAPP   LUN              8010  /dev/sdaq  /dev/sg42
[3:0:0:10]   disk    NETAPP   LUN              8010  /dev/sdar  /dev/sg43
[3:0:0:11]   disk    NETAPP   LUN              8010  /dev/sdas  /dev/sg44
[3:0:0:12]   disk    NETAPP   LUN              8010  /dev/sdat  /dev/sg45
[3:0:0:13]   disk    NETAPP   LUN              8010  /dev/sdau  /dev/sg46
[3:0:0:14]   disk    NETAPP   LUN              8010  /dev/sdav  /dev/sg47
[3:0:0:15]   disk    NETAPP   LUN              8010  /dev/sdaw  /dev/sg48
[3:0:0:16]   disk    NETAPP   LUN              8010  /dev/sdax  /dev/sg49
```

Alternatively, the LUNs discovered by the Chelsio FCoE initiators can be accessed via easily-identifiable ‘udev’ path device files like:

```
[root@host~]# ls /dev/disk/by-path/pci-0000:04:00.0-csio-fcoe
<local_wwpn>:<remote_wwpn>:<lun_wwn>
```

```
[root@temi ~]# mount /dev/disk/by-path/pci-0000:03:00.6-csio-fcoe-0x50007430463b7380:0x500a0981892bb831:0x0000000000000000 /mnt/
[root@temi ~]# mount
/dev/sdbe5 on / type ext4 (rw)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
devpts on /dev/pts type devpts (rw,gid=5,mode=620)
tmpfs on /dev/shm type tmpfs (rw)
/dev/sdbe1 on /boot type ext3 (rw)
none on /proc/sys/fs/binfmt_misc type binfmt_misc (rw)
/tmp on /tmp type none (rw,bind)
/var/tmp on /var/tmp type none (rw,bind)
/home on /home type none (rw,bind)
sunrpc on /var/lib/nfs/rpc_pipefs type rpc_pipefs (rw)
none on /sys/kernel/debug type debugfs (rw)
gvfs-fuse-daemon on /root/.gvfs type fuse.gvfs-fuse-daemon (rw,nosuid,nodev)
/dev/sdah on /mnt type ext3 (rw)
[root@temi ~]#
```

4.4. Creating Filesystem

Create an ext3 filesystem using the following command:

```
[root@host~]# mkfs.ext3 /dev/sdx
```

```
[root@temi ~]# mkfs.ext3 /dev/sdah
mke2fs 1.41.12 (17-May-2010)
/dev/sdah is entire device, not just one partition!
Proceed anyway? (y,n) y
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=1 blocks, Stripe width=16 blocks
327680 inodes, 1310720 blocks
65536 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=1342177280
40 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736

Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 25 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
```

4.5. Mounting the formatted LUN

The formatted LUN can be mounted on the specified mountpoint using the following command:

```
[root@host~]# mount /dev/sdx /mnt
```

```
[root@temi ~]# mount /dev/sdah /mnt/
[root@temi ~]# mount
/dev/sdbo5 on / type ext4 (rw)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
devpts on /dev/pts type devpts (rw,gid=5,mode=620)
tmpfs on /dev/shm type tmpfs (rw)
/dev/sdbo1 on /boot type ext3 (rw)
none on /proc/sys/fs/binfmt_misc type binfmt_misc (rw)
/tmp on /tmp type none (rw,bind)
/var/tmp on /var/tmp type none (rw,bind)
/home on /home type none (rw,bind)
sunrpc on /var/lib/nfs/rpc_pipefs type rpc_pipefs (rw)
none on /sys/kernel/debug type debugfs (rw)
gvfs-fuse-daemon on /root/.gvfs type fuse.gvfs-fuse-daemon (rw,nosuid,nodev)
/dev/sdah on /mnt type ext3 (rw)
```

XIV. Offload Bonding driver

1. Introduction

The Chelsio Offload bonding driver provides a method to aggregate multiple network interfaces into a single logical bonded interface effectively combining the bandwidth into a single connection. It also provides redundancy in case one of link fails.

The traffic running over the bonded interface can be fully offloaded to the adapter, thus freeing the CPU from TCP/IP overhead.

1.1. Hardware Requirements

1.1.1. Supported adapters

The following are the currently shipping Chelsio adapters that are compatible with the Chelsio Offload Bonding driver:

- T62100-LP-CR
- T6225-CR
- T520-BT
- T580-CR
- T520-LL-CR
- T520-CR
- T580-LP-CR
- T540-CR
- T420-CR
- T440-CR
- T422-CR
- T420-SO-CR
- T404-BT
- T420-BCH
- T440-LP-CR
- T420-BT
- T420-LL-CR
- T420-CX

1.2. Software Requirements

1.2.1. Linux Requirements

Currently the Offload Bonding driver is available for the following versions:

- RHEL 7.3, 3.10.0-514.el7
- RHEL 7.3, 3.10.0-514.el7.ppc64le (POWER8 LE)
- RHEL 7.3, 4.5.0-15.el7.aarch64 (ARM64)

- RHEL 7.2, 3.10.0-327.el7
- RHEL 6.8, 2.6.32-642.el6
- RHEL 6.7, 2.6.32-573.el6
- SLES 12 SP2, 4.4.21-69-default
- SLES 12 SP1, 3.12.49-11-default
- SLES 11 SP4, 3.0.101-63-default
- Ubuntu 16.04.1, 4.4.0-47-generic
- Ubuntu 14.04.4, 4.2.0-27-generic
- Kernel.org linux-4.8

Other kernel versions have not been tested and are not guaranteed to work.

2. Software/Driver Loading

Important *Please ensure that all inbox drivers are unloaded before proceeding with unified wire drivers.*

The driver must be loaded by the root user. Any attempt to load the driver as a regular user will fail.

To load the driver (with offload support), run the following command:

```
[root@host~]# modprobe bonding
```

3. Software/Driver Unloading

To unload the driver, run the following command:

```
[root@host~]# rmmod bonding
```

4. Software/Driver Configuration and Fine-tuning

4.1. Offloading TCP traffic over a bonded interface

The Chelsio Offload Bonding driver supports all the bonding modes in NIC Mode. In offload mode (t4_tom loaded) however, only the **balance-rr (mode=0)**, **active-backup (mode=1)**, **balance-xor (mode=2)** and **802.3ad (mode=4)** modes are supported.

To offload TCP traffic over a bonded interface, use the following method:

i. Load the network driver with TOE support:

```
[root@host~]# modprobe t4_tom
```

ii. Create a bonded interface:

```
[root@host~]# modprobe bonding mode=1 miimon=100
```

iii. Bring up the bonded interface and enslave the interfaces to the bond:

```
[root@host~]# ifconfig bond0 up
[root@host~]# ifenslave bond0 ethX ethY
```

Note *__ethX__ and __ethY__ are interfaces of the same adapter.*

iv. Assign IPv4/IPv6 address to the bonded interface:

```
[root@host~]# ifconfig bond0 X.X.X.X/Y
[root@host~]# ifconfig bond0 inet6 add <128-bit IPv6 Address> up
```

v. Disable FRTO on the PEER:

```
[root@host~]# sysctl -w net.ipv4.tcp_frto=0
```

All TCP traffic will be offloaded over the bonded interface now.

XV. Offload Multi-adapter Failover (MAFO)

1. Introduction

Chelsio's T5 and T4-based adapters offer a complete suite of high reliability features, including adapter-to-adapter failover. The patented offload Multi-adapter Failover (MAFO) feature ensures all offloaded traffic continue operating seamless in the face of port failure.

MAFO allows aggregating network interfaces across multiple adapters into a single logical bonded interface, providing effective fault tolerance.

The traffic running over the bonded interface can be fully offloaded to the adapter, thus freeing the CPU from TCP/IP overhead.

Important

- *Portions of this software are covered under US Patent, Failover and migration for full-offload network interface devices : US 8346919 B1*
- *Use of the covered technology is strictly limited to Chelsio ASIC-based soutions.*

1.1. Hardware Requirements

1.1.1. Supported adapters

The following are the currently shipping Chelsio adapters that are compatible with the Offload Multi-adapter Failover driver.

- T520-BT
- T580-CR
- T520-LL-CR
- T520-CR
- T580-LP-CR
- T540-CR
- T420-CR
- T440-CR
- T422-CR
- T420-SO-CR
- T404-BT
- T420-BCH
- T440-LP-CR
- T420-BT
- T420-LL-CR
- T420-CX

1.2. Software Requirements

1.2.1. Linux Requirements

Currently the Offload Multi-adapter Failover driver is available for the following versions:

- RHEL 7.3, 3.10.0-514.el7
- RHEL 7.3, 3.10.0-514.el7.ppc64le (POWER8 LE)
- RHEL 7.3, 4.5.0-15.el7.aarch64 (ARM64)
- RHEL 7.2, 3.10.0-327.el7
- RHEL 6.8, 2.6.32-642.el6
- RHEL 6.7, 2.6.32-573.el6
- SLES 12 SP2, 4.4.21-69-default
- SLES 12 SP1, 3.12.49-11-default
- SLES 11 SP4, 3.0.101-63-default
- Ubuntu 16.04.1, 4.4.0-47-generic
- Ubuntu 14.04.4, 4.2.0-27-generic
- Kernel.org linux-4.8

Other kernel versions have not been tested and are not guaranteed to work.

1.2.2. Driver Requirements

Multi-adapter failover feature will work for *Link Down* events caused by:

- Cable unplug on bonded interface.
- Bringing corresponding switch port down.

Note *The feature will not work if the bonded interfaces are administratively taken down.*

2. Software/Driver Loading

Important *Please ensure that all inbox drivers are unloaded before proceeding with unified wire drivers.*

The driver must be loaded by the root user. Any attempt to load the driver as a regular user will fail.

To load the driver (with offload support), run the following command:

```
[root@host~]# modprobe bonding
```

3. Software/Driver Unloading

To unload the driver, run the following command:

```
[root@host~]# rmmod bonding
```

4. Software/Driver Configuration and Fine-tuning

4.1. Offloading TCP traffic over a bonded interface

The Chelsio MAFO driver supports only the **active-backup (mode=1)** mode.

To offload TCP traffic over a bonded interface, use the following method:

i. Load the network driver with TOE support:

```
[root@host~]# modprobe t4_tom
```

ii. Create a bonded interface:

```
[root@host~]# modprobe bonding mode=1 miimon=100
```

iii. Bring up the bonded interface and enslave the interfaces to the bond:

```
[root@host~]# ifconfig bond0 up
[root@host~]# ifenslave bond0 ethX ethY
```

Note *__ethX__ and __ethY__ are interfaces of different adapters.*

iv. Assign IPv4/IPv6 address to the bonded interface

```
[root@host~]# ifconfig bond0 X.X.X.X/Y
[root@host~]# ifconfig bond0 inet6 add <128-bit IPv6 Address> up
```

vi. Disable FRTO on the PEER:

```
[root@host~]# sysctl -w net.ipv4.tcp_frto=0
```

All TCP traffic will be offloaded over the bonded interface now and fail-over will happen in case of link-down event.

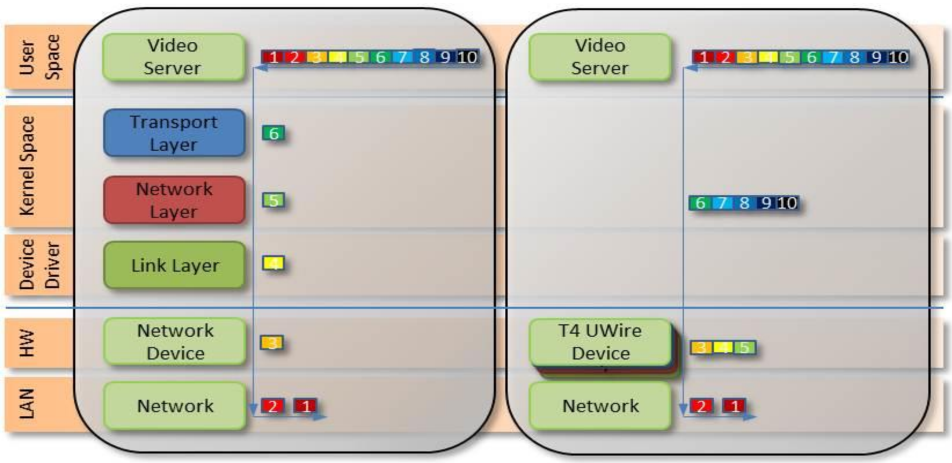
XVI. UDP Segmentation Offload and Pacing

1. Introduction

Chelsio's T5/T4 series of adapters provide UDP segmentation offload and per-stream rate shaping to drastically lower server CPU utilization, increase content delivery capacity, and improve service quality.

Tailored for UDP content, UDP Segmentation Offload (USO) technology moves the processing required to packetize UDP data and rate control its transmission from software running on the host to the network adapter. USO increases performance and dramatically reduces CPU overhead, allowing significantly higher capacity using the same server hardware. Without USO support, UDP server software running on the host needs to packetize payload into frames, process each frame individually through the network stack and schedule individual frame transmission, resulting in millions of system calls, and packet traversals through all protocol layers in the operating system to the network device. In contrast, USO implements the network protocol stack in the adapter, and the host server software simply hands off unprocessed UDP payload in large I/O buffers to the adapter.

The following figure compares the traditional datapath on the left to the USO datapath on the right, showing how per-frame processing is eliminated. In this example, the video server pushes 5 frames at a time. In an actual implementation, a video server pushes 50 frames or more in each I/O, drastically lowering the CPU cycles required to deliver the content.



Pacing is beneficial for several reasons, one example is for Content Delivery Networks (CDNs)/Video On Demand (VOD) providers to avoid receive buffer overflows, smooth out network traffic, or to enforce Service Level Agreements (SLAs).

1.1. Hardware Requirements

1.1.1. Supported adapters

The following are the currently shipping Chelsio adapters that are compatible with the UDP Segmentation Offload and Pacing driver.

- T520-BT
- T580-CR
- T520-LL-CR
- T520-CR
- T580-LP-CR
- T540-CR
- T420-CR
- T440-CR
- T422-CR
- T404-BT
- T420-BCH
- T440-LP-CR
- T420-BT
- T420-LL-CR
- T420-CX

1.2. Software Requirements

1.2.1. Linux Requirements

Currently the UDP Segmentation Offload and Pacing driver is available for the following versions:

- RHEL 7.3, 3.10.0-514.el7
- RHEL 7.2, 3.10.0-327.el7
- RHEL 6.8, 2.6.32-642.el6
- RHEL 6.7, 2.6.32-573.el6
- SLES 12 SP2, 4.4.21-69-default
- SLES 12 SP1, 3.12.49-11-default
- SLES 11 SP4, 3.0.101-63-default
- Ubuntu 16.04.1, 4.4.0-47-generic
- Ubuntu 14.04.4, 4.2.0-27-generic
- Kernel.org linux-4.8

Other kernel versions have not been tested and are not guaranteed to work.

2. Software/Driver Loading

Important *Please ensure that all inbox drivers are unloaded before proceeding with unified wire drivers.*

The driver must be loaded by the root user. Any attempt to load the driver as a regular user will fail.

Run the following commands to load the driver:

```
[root@host~]# modprobe cxgb4
[root@host~]# modprobe t4_tom
```

Though normally associated with the Chelsio TCP Offload engine, the *t4_tom* module is required in order to allow for the proper redirection of UDP socket calls.

3. Software/Driver Unloading

Reboot the system to unload the driver. To unload without rebooting, refer **Unloading the TOE driver** section of **Network (NIC/TOE)** chapter.

4. Software/Driver Configuration and Fine-tuning

4.1. Modifying the application

To use the UDP offload functionality, the application needs to be modified. Follow the steps mentioned below:

i. Determine the UDP socket file descriptor in the application through which data is sent

ii. Declare and initialize two variables in the application:

```
int fs=1316;
int cl=1;
```

Here,

- *fs* is the UDP packet payload size in bytes that is transmitted on the wire. The minimum value of fs is 256 bytes.
- *cl* is the UDP traffic class(scheduler-class-index) that the user wishes to assign the data stream to. This value needs to be in the range of 0 to 14 for T4 series of adapters and a range of 0 to 15 for T5 series of adapters.

The application will function according to the parameters set for that traffic class.

iii. Add socket option definitions:

In order to use *setsockopt()* to set the options to the UDP socket, the following three definitions need to be made:

- SO_FRAMESIZE used for setting frame size, which has the value 291.
- SOL_SCHEDCLASS used for setting UDP traffic class, which has the value 290.
- IPPROTO_UDP used for setting the type of IP Protocol.

```
# define SO_FRAMESIZE 291
# define SOL_SCHEDCLASS 290
# define IPPROTO_UDP 17
```

iv. Use the *setsockopt()* function to set socket options:

```
//Get the UDP socket descriptor variable
setsockopt (sockfd , IPPROTO_UDP, SO_FRAMESIZE, &fs, sizeof(fs));
setsockopt (sockfd , IPPROTO_UDP, SOL_SCHEDCLASS, &cl, sizeof(cl));
```

Here:

- *sockfd* : The file descriptor of the UDP socket
- *&fs* / *&cl* : Pointer to the framesize and class variables
- *sizeof(fs)* / *sizeof(cl)* : The size of the variables

v. Now, compile the application.

4.1.1. UDP offload functionality for RTP data

In case of RTP data, the video server application sends the initial sequence number and the RTP payload. The USO engine segments the payload data, increments the sequence number and sends out the data.

In order to use the UDP offload functionality for RTP data, make the following additions to the steps mentioned above:

1. In step (ii), declare and initialize a new variable in the application:

```
int rtp_header_size=16;
```

Here, *rtp_header_size* is the RTP header size in bytes that the application sends.

2. In step (iii), define a new macro, *UDP_RTPHEADERLEN* used for setting RTP header length with the value 292.

```
# define UDP_RTPHEADERLEN 292
```

3. In step (iv), define a new socket option:

```
setsockopt (sockfd,17,UDP_RTPHEADERLEN,&rtp_header_size,
sizeof(rtp_header_size));
```

Here,

- *&rtp_header_size* : pointer to the RTP header length variable
- *sizeof(rtp_header_size)* : the size of the RTP header length variable

4.2. Configuring UDP Pacing

Now that the application has been modified to associate the application's UDP socket to a particular UDP traffic class, the pacing of that socket's traffic can be set using the *cxgbtool* utility.

i. Bring up the network interface:

```
[root@host~]# ifconfig <ethX> up
```

ii. Run the following command:

```
[root@host~]# cxgbtool <ethX> sched-class params  type packet  level cl-rl
mode flow  rate-unit bits  rate-mode absolute channel <Channel No.> class
<scheduler-class-index> max-rate <maximum-rate>  pkt-size <Packet size>
```

Here,

- *ethX* is the Chelsio interface
- *Channel No.* is the port on which data is flowing (0-3)
- *scheduler-class-index* is the UDP traffic class (0-14 for T4 series of adapters and 0-15 for T5 series of adapters) set in the SOL_SCHEDCLASS socket option in the application in section 4.1.
- *maximum-rate* is the bit rate (Kbps) for this UDP stream. This value should be in the range of 50 Kbps to 50 Mbps for T4 adapters. For T5 adapters, the value should be in the range of 100 kbps to 1 Gbps.
- *Packet size* is the UDP packet payload size in bytes; it should be equal to the value set in the *SO_FRAMESIZE* socket option in the application in section 4.1.

Example:
The user wants to transfer UDP data on port 0 of the adapter using the USO engine. The application has been modified as shown in section 4.1. In order to set a bit rate of 10Mbps for traffic class 1 with payload size of 1316 on port 0, the following invocation of *cxgbtool* is used:

```
[root@host~]# cxgbtool ethX sched-class params  type packet  level cl-rl
mode flow  rate-unit bits  rate-mode absolute channel 0  class 1 max-rate
10000  pkt-size 1316
```

Note *To get an accurate bit rate per class, data sent by the application to the sockets should be a multiple of the value set for the "pkt-size" parameter. In above example, IO size sent by application should be a multiple of 1316.*

Note *Linux Unified Wire currently supports 10240 offload UDP connections. If the application needs to establish more than 10240 UDP connections, it can check the return code of ENOSPC from a send() or sendto() call and close this socket and open a new one that uses the kernel UDP stack.*

XVII. Offload IPv6 driver

1. Introduction

The growth of the Internet has created a need for more addresses than are possible with IPv4. Internet Protocol version 6 (IPv6) is a version of the Internet Protocol (IP) designed to succeed the Internet Protocol version 4 (IPv4).

Chelsio's Offload IPv6 feature provides support to fully offload IPv6 traffic to the Unified Wire adapter.

1.1. Hardware Requirements

1.1.1. Supported adapters

The following are the currently shipping Chelsio adapters that are compatible with Chelsio Offload IPv6 feature:

- T62100-LP-CR
- T6225-CR
- T520-BT
- T580-CR
- T520-LL-CR
- T520-CR
- T580-LP-CR
- T540-CR
- T420-CR
- T440-CR
- T422-CR
- T420-SO-CR
- T404-BT
- T420-BCH
- T440-LP-CR
- T420-BT
- T420-LL-CR
- T420-CX

1.2. Software Requirements

1.2.1. Linux Requirements

Currently the Offload IPv6 feature is available for the following versions:

- RHEL 7.3, 3.10.0-514.el7
- RHEL 7.3, 3.10.0-514.el7.ppc64le (POWER8 LE)
- RHEL 7.3, 4.5.0-15.el7.aarch64 (ARM64)

- RHEL 7.2, 3.10.0-327.el7
- RHEL 6.8, 2.6.32-642.el6
- RHEL 6.7, 2.6.32-573.el6
- SLES 12 SP2, 4.4.21-69-default
- SLES 12 SP1, 3.12.49-11-default
- SLES 11 SP4, 3.0.101-63-default
- Ubuntu 16.04.1, 4.4.0-47-generic
- Ubuntu 14.04.4, 4.2.0-27-generic
- Kernel.org linux-4.8

Other kernel versions have not been tested and are not guaranteed to work.

2. Software/Driver Loading

Important *Please ensure that all inbox drivers are unloaded before proceeding with unified wire drivers.*

IPv6 must be enabled in your system (enabled by default) to use the Offload IPv6 feature.Also, Unified Wire package must be installed with IPv6 support (see Software/Driver Installation).

After installing Unified Wire package and rebooting the host, load the NIC (*cxgb4*) and TOE (*t4_tom*) drivers. The drivers must be loaded by the root user. Any attempt to load the drivers as a regular user will fail.

```
[root@host~]# modprobe cxgb4
[root@host~]# modprobe t4_tom
```

All the IPv6 traffic will be offloaded now.

3. Software/Driver Unloading

To disable Offload IPv6 feature, unload NIC and TOE drivers:

3.1. Unloading the NIC driver

To unload the NIC driver, run the following command:

```
[root@host~]# rmmod cxgb4
```

3.2. Unloading the TOE driver

Please reboot the system to unload the TOE driver. To unload without rebooting, refer **Unloading the TOE driver** section of **Network (NIC/TOE)** chapter.

XVIII. WD Sniffing and Tracing

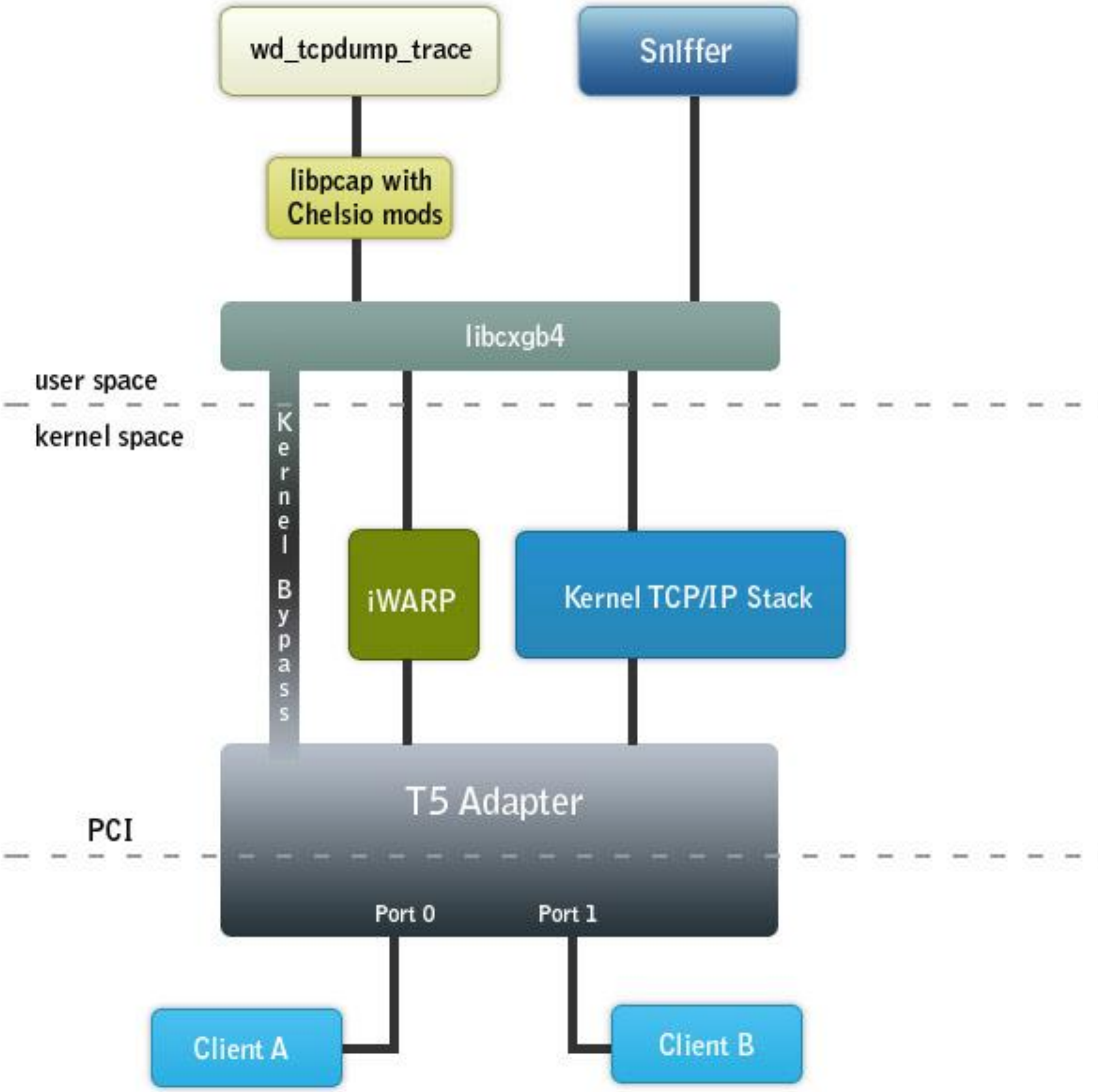
1. Theory of Operation

The objective of these utilities (*wd_sniffer* and *wd_tcpdump_trace*) is to provide sniffing and tracing capabilities by making use of T5/T4's hardware features.

- Sniffer is a tool to measure bandwidth and involves targeting specific multicast traffic and sending it directly to user space.
  a) Get a Queue (raw QP) idx.
  b) Program a filter to redirect specific traffic to the raw QP queue.

- Tracer - All tapped traffic is forwarded to user space and also pushed back on the wire via the internal loop back mechanism
  a) Get a Queue (raw QP) idx
  b) Set the T4 adapter in loop back
  c) Connect Client A and B to ports 0 and 1 or ports 2 and 3.
  d) Enable tracing.

In either mode the targeted traffic bypasses the kernel TCP/IP stack and is delivered directly to user space by means of an RX queue.



Schematic diagram of sniffer and tracer

1.1. Hardware Requirements

1.1.1. Supported adapters

The following are the currently shipping Chelsio adapters that are compatible with the tools:

- T520-BT
- T580-CR
- T580-LP-CR
- T520-LL-CR
- T520-CR

- T540-CR
- T420-CR
- T440-CR
- T422-CR
- T404-BT
- T420-BCH
- T440-LP-CR
- T420-BT
- T420-LL-CR
- T420-CX

1.2. Software Requirements

1.2.1. Linux Requirements

Currently the WD Sniffing and Tracing utility is available for the following version:

- RHEL 7.3, 3.10.0-514.el7
- RHEL 7.2, 3.10.0-327.el7
- RHEL 6.8, 2.6.32-642.el6
- RHEL 6.7, 2.6.32-573.el6
- SLES 12 SP2, 4.4.21-69-default
- SLES 12 SP1, 3.12.49-11-default
- SLES 11 SP4, 3.0.101-63-default
- Ubuntu 16.04.1, 4.4.0-47-generic
- Ubuntu 14.04.4, 4.2.0-27-generic
- Kernel.org linux-4.8

Other kernel versions have not been tested and are not guaranteed to work.

2. Installation and Usage

2.1. Installing basic support

iw_cxgb4 (Chelsio iWARP driver) and *cxgb4* (Chelsio NIC driver) drivers have to be compiled and loaded before running the utilities. Refer to the **Software/Driver Loading** section for each driver and follow the instructions mentioned before proceeding.

2.2. Using Sniffer (wd_*sniffer)*

1. **Setup**:

Wire filter sniffing requires 2 systems with one machine having a T5/T4 card.

The machines should be setup in the following manner:

Machine A <---------> Machine B

192.168.1.100 192.168.1.200

2. **Procedure**:

On the Device Under Test (DUT), start sniffer.

```
[root@host~]# wd_sniffer -T 20 -s 1000 -I <MAC address of interface to
sniff>
```

Start traffic on the PEER and watch the sniffer.

The sniffer will receive all packets as fast as possible, update the packet count, and then discard the data. Performance is a full 10Gbps for packet size 1000.

2.3. Using Tracer (wd_tcpdump_trace)

1. **Setup**:

Wire tapping requires 3 systems with one machine having a T5/T4 card with two or more ports. The machines should be setup in the following manner:

DUT: Machine B

PEER: Machine A <-----> (port 0) (port 1) <-----> PEER: Machine C

192.168.1.100 IP-dont-care IP-dont-care 192.168.1.200

2. Procedure:

Run `wd_tcpdump_trace -i iface` on the command prompt where *iface* is one of the interfaces whose traffic you want to trace. In the above diagram its port 0 or port 1.

```
[root@host~]# wd_tcpdump_trace -i <iface>
```

Use any tool (like ping or ssh) to run traffic between machines A and B. The traffic should successfully make it from end to end and wd_tcpdump_trace on the DUT should show the tapped traffic.

XIX. Classification and Filtering

1. Introduction

Classification and Filtering feature enhances network security by controlling incoming traffic as they pass through network interface based on source and destination addresses, protocol, source and receiving ports, or the value of some status bits in the packet. This feature can be used in the ingress path to:

- Steer ingress packets that meet ACL (Access Control List) accept criteria to a particular receive queue.
- Switch (proxy) ingress packets that meet ACL accept criteria to an output port, with optional header rewrite.
- Drop ingress packets that fail ACL accept criteria.

1.1. Hardware Requirements

1.1.1. Supported adapters

The following are the currently shipping Chelsio adapters that are compatible with the Classification and Filtering feature:

- T62100-LP-CR
- T6225-CR
- T520-BT
- T580-CR
- T520-LL-CR
- T520-SO-CR
- T520-CR
- T540-CR
- T580-LP-CR
- T580-SO-CR
- T420-CR
- T440-CR
- T422-CR
- T420-SO-CR
- T404-BT
- T420-BCH
- T440-LP-CR
- T420-BT
- T420-LL-CR
- T420-CX

1.2. Software Requirements

1.2.1. Linux Requirements

Currently the Classification and Filtering feature is available for the following versions:

- RHEL 7.3, 3.10.0-514.el7
- RHEL 7.3, 3.10.0-514.el7.ppc64le (POWER8 LE)
- RHEL 7.3, 4.5.0-15.el7.aarch64 (ARM64)
- RHEL 7.2, 3.10.0-327.el7
- RHEL 6.8, 2.6.32-642.el6
- RHEL 6.7, 2.6.32-573.el6
- SLES 12 SP2, 4.4.21-69-default
- SLES 12 SP1, 3.12.49-11-default
- SLES 11 SP4, 3.0.101-63-default
- Ubuntu 16.04.1, 4.4.0-47-generic
- Ubuntu 14.04.4, 4.2.0-27-generic
- Kernel.org linux-4.8

Other kernel versions have not been tested and are not guaranteed to work.

2. Usage

2.1. Configuration

The Classification and Filtering feature is configured by specifying the filter selection combination set in the firmware configuration located in */lib/firmware/cxgb4/*

The following combination is set by default and packets will be matched accordingly:

- For T5/T6:

```
filterMode = fcoemask, srvrsram, fragmentation, mpshittype, protocol, vlan,
port, fcoe
```

- For T4:

```
filterMode = fragmentation, mpshittype, protocol, vlan, port, fcoe
```

Where,

srvrsram : server-sram
fragmentation: Fragmented IP packets
mpshittype : MAC address "match type" (0=unicast, 1=unicast hash, 2=multicast, 3=multicast hash, 4=PROM, 5=hyper PROM, 6=broadcast, 7=none)
protocol : IP protocol number (ICMP=1, TCP=6, UDP=17, etc)
vlan : Inner VLAN Tag
port : Packet ingress port number
fcoe : Fibre Channel over Ethernet frames

2.2. Creating Filter Rules

Network driver (*cxgb4*) must be installed and loaded before setting the filter rule.

i. If you haven't done already, run the Unified Wire Installer with the appropriate configuration tuning option to install the Network Driver.
ii. Next, run the following command to load the network driver:

```
[root@host~]# modprobe cxgb4
```

iii. Now, create filter rules using *cxgbtool*:

```
[root@host~]# cxgbtool ethx filter <index> action [pass/drop/switch]
```

Where,

ethX : Chelsio interface
index : positive integer set as filter id
action : Ingress packet disposition
pass : Ingress packets will be passed through set ingress queues
switch : Ingress packets will be routed to an output port with optional header rewrite.
drop : Ingress packets will be dropped.

Note *In case of multiple filter rules, the rule with the lowest filter index takes higher priority.*

2.2.1. Examples

- ***Drop* action**

```
[root@host~]# cxgbtool ethX filter 0 action drop fip 192.168.1.5
```

The above filter rule will drop all ingress packets from IP 192.168.1.5

- ***Pass* action**

```
[root@host~]# cxgbtool ethX filter 0 action pass lport 10001 fport 355
queue 2
```

The above filter rule will pass all ingress packets that match local port 10001 and remote port 355 to ingress queue 2 for load balancing.

- ***Switch* action**

```
[root@host~]# cxgbtool ethX filter 0 action switch iport 0 eport 1 vlan 3
```

The above filter rule will route all ingress packets that match VLAN id 3 from port 0 of Chelsio adapter to port 1. Remaining packets will be sent to the host.

For offloaded ingress packets, use the `prio` argument with the above command:

```
[root@host~]# cxgbtool ethx filter <index> action <pass/drop/switch> prio 1
```

Note

- *For more information on additional parameters, refer cxgbtool manual by running the `man cxgbtool` command.*
- `prio` *argument is not supported for LE-TCAM filters when T5/T6 Hash Filter config file is used.*

2.3. Listing Filter Rules

To list the filters set, run the following command:

```
[root@host~]# cxgbtool ethX filter show
```

2.4. Removing Filter Rules

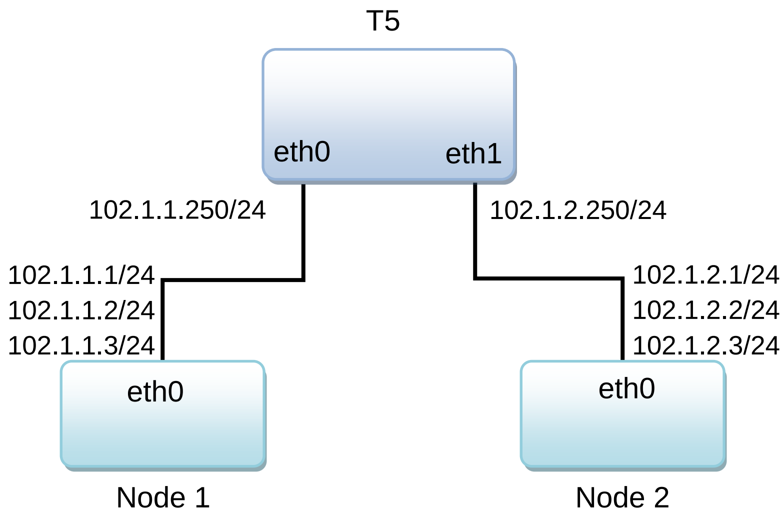
To remove a filter, run the following command with the corresponding filter rule index

```
[root@host~]# cxgbtool ethX filter index <delete|clear>
```

Note *For more information on additional parameters, refer cxgbtool manual by running the `man cxgbtool` command*

2.5. Layer 3 example

Here's an example on how to achieve L3 routing functionality:



- **Follow these steps on Node 1**

i. Configure IP address and enable the 3 interfaces:

```
[root@host~]# ifconfig eth0 102.1.1.1/24 up
[root@host~]# ifconfig eth0:2 102.1.1.2/24 up
[root@host~]# ifconfig eth0:3 102.1.1.3/24 up
[root@host~]# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:07:43:04:7D:50
          inet addr:102.1.1.1  Bcast:102.1.1.255  Mask:255.255.255.0
          inet6 addr: fe80::207:43ff:fe04:7d50/64 Scope:Link
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:14372 errors:0 dropped:0 overruns:0 frame:0
          TX packets:62203 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1585952 (1.5 MiB)  TX bytes:4798122 (4.5 MiB)
          Interrupt:16
```

ii. Setup a static OR default route towards T5 router to reach 102.1.2.0/24 network

```
[root@host~]# route add -net 102.1.2.0/24 gw 102.1.1.250
```

- **Follow these steps on Node 2**

i. Configure IP address and enable the 3 interfaces:

```
[root@host~]# ifconfig eth0 102.1.2.1/24 up
[root@host~]# ifconfig eth0:2 102.1.2.2/24 up
[root@host~]# ifconfig eth0:3 102.1.2.3/24 up
[root@host~]# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:07:43:12:D4:88
          inet addr:102.1.2.1  Bcast:102.1.2.255  Mask:255.255.255.0
          inet6 addr: fe80::7:4300:112:d488/64 Scope:Link
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:1961 errors:0 dropped:2 overruns:0 frame:0
          TX packets:141 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:218606 (213.4 KiB)  TX bytes:17483 (17.0 KiB)
          Interrupt:17
```

ii. Setup a static OR default route towards T5 router to reach 102.1.1.0/24 network

```
[root@host~]# route add -net 102.1.1.0/24 gw 102.1.2.250
```

- **Follow these steps on machine with T5 adapter**

i. Configure IP address and enable the 2 interfaces:

```
[root@host~]# ifconfig eth0 102.1.1.250/24 up
[root@host~]# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:07:43:04:96:40
          inet addr:102.1.1.250  Bcast:102.1.1.255  Mask:255.255.255.0
          inet6 addr: fe80::207:43ff:fe04:9640/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:114 errors:0 dropped:0 overruns:0 frame:0
          TX packets:535 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:11880 (11.6 KiB)  TX bytes:61729 (60.2 KiB)
          Interrupt:16
[root@host~]# ifconfig eth1 102.1.2.250/24 up
[root@host~]# ifconfig eth1
eth1      Link encap:Ethernet  HWaddr 00:07:43:04:96:48
          inet addr:102.1.2.250  Bcast:102.1.2.255  Mask:255.255.255.0
          inet6 addr: fe80::7:4300:104:9648/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:31 errors:0 dropped:0 overruns:0 frame:0
          TX packets:433 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3181 (3.1 KiB)  TX bytes:49134 (47.9 KiB)
          Interrupt:16
```

ii. Create filter rule to send packets for 102.1.2.0/24 network out via eth1 interface:

```
[root@host~]# cxgbtool eth0 filter 0 lip 102.1.2.0/24 hitcnts 1 action
switch  eport 1 smac 00:07:43:04:96:48 dmac 00:07:43:12:D4:88
```

Where, `smac` is the MAC address of eth1 interface on T5 adapter machine and `dmac` is the MAC address of eth0 interface on Node 2.

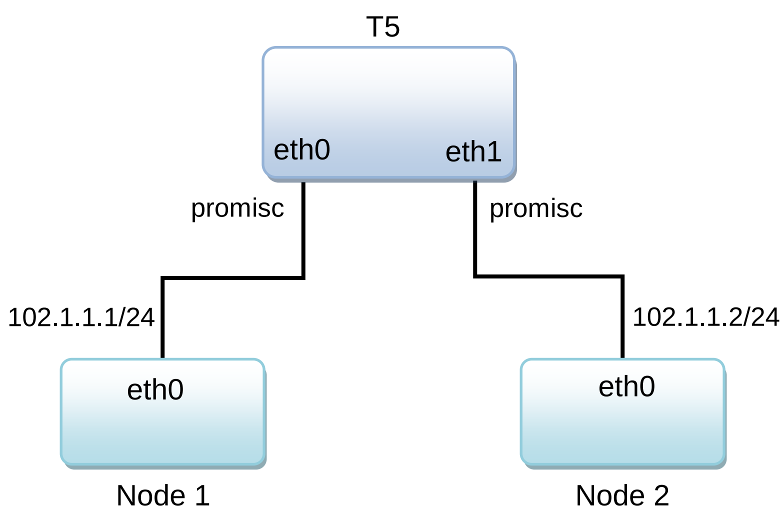
iii. Create filter rule to send packets for 102.1.1.0/24 network out via eth0 inteface

```
[root@host~]# cxgbtool eth0 filter 1 lip 102.1.1.0/24 hitcnts 1 action
switch eport 0 smac 00:07:43:04:96:40 dmac 00:07:43:04:7D:50
```

Where, `smac` is the MAC address of eth0 interface on T5 adapter machine and `dmac` is the MAC address of eth0 interface on Node 1.

2.6. Layer 2 example

Here's an example on how to achieve L2 switching functionality. The following will only work on kernel 3.10 and above.



- **Follow these steps on Node 1**

i. Configure IP address and enable the interface:

```
[root@host~]# ifconfig eth0 102.1.1.1/24 up
[root@host~]# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:07:43:04:7D:50
          inet addr:102.1.1.1  Bcast:102.1.1.255  Mask:255.255.255.0
          inet6 addr: fe80::207:43ff:fe04:7d50/64 Scope:Link
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:14372 errors:0 dropped:0 overruns:0 frame:0
          TX packets:62203 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1585952 (1.5 MiB)  TX bytes:4798122 (4.5 MiB)
          Interrupt:16
```

ii. Setup ARP entry to reach 102.1.1.2

```
[root@host~]# arp -s 102.1.1.2 00:07:43:12:D4:88
```

- **Follow these steps on Node 2**

i. Configure IP address and enable the interface:

```
[root@host~]# ifconfig eth0 102.1.1.2/24 up
[root@host~]# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:07:43:12:D4:88
          inet addr:102.1.1.2  Bcast:102.1.1.255  Mask:255.255.255.0
          inet6 addr: fe80::7:4300:112:d488/64 Scope:Link
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:1961 errors:0 dropped:2 overruns:0 frame:0
          TX packets:141 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:218606 (213.4 KiB)  TX bytes:17483 (17.0 KiB)
          Interrupt:17
```

ii. Setup ARP entry to reach 102.1.1.1

```
[root@host~]# arp -s 102.1.1.1 00:07:43:04:7D:50
```

- **Follow these steps on machine with T5 adapter**

i. Update filtermode value with below combination in */lib/firmware/cxgb4/t5-config.txt* to enable matching based on macidx

```
filterMode = fragmentation, macmatch, mpshittype, protocol, tos, port, fcoe
```

ii. Unload and re-load the *cxgb4* driver.

iii. Enable promiscuous mode on both the interfaces on T5 adapter machine:

```
[root@host~]# ifconfig eth0 up promisc
[root@host~]# ifconfig eth1 up promisc
```

iv. Build and install latest iproute2 package

v. Add fdb entry corresponding to Node-2 on T5's eth0 interface:

```
[root@host~]# bridge fdb add 00:07:43:12:D4:88 dev eth0 self
```

vi. Add fdb entry corresponding to Node-1 on T5's eth1 interface:

```
[root@host~]# bridge fdb add 00:07:43:04:7D:50 dev eth1 self
```

vii. Both MAC entries should show up in MPS table. Run the following command to view the table and note the index (*idx* field) of the entries:

```
[root@host~]# cat /sys/kernel/debug/cxgb4/0000\:01\:00.4/mps_tcam | more
```

viii. Create a filter to match incoming packet's *dst-mac 00:07:43:12:d4:88* with particular mac-idx and switch it out via eport 1:

```
[root@host~]# cxgbtool eth0 filter 0 macidx 5 action switch eport 1 hitcnts 1
```

ix. Create a filter to match incoming packet's *dst-mac 00:07:43:04:7d:50* with particular mac-idx and switch it out via eport 0:

```
[root@host~]# cxgbtool eth0 filter 1 macidx 7 action switch eport 0 hitcnts
1
```

3. Hash/DDR Filters

The default (*Unified Wire)* configuration tuning option allows you to create LE-TCAM filters, which has a limit of 496 filter rules. If you wish to create more, select *T5/T6 Hash Filter* configuration tuning option during installation which allows you to create HASH/DDR filters with a capacity of ~0.5 million filter rules.

Note *Creating Hash/DDR Filters is currently supported only on T5/T6 adapters.*

3.1. Creating Filter Rules

Network driver (*cxgb4*) must be installed and loaded before setting the filter rule.

i. If you haven't done already, run the Unified Wire Installer with the *T5/T6 Hash Filter* configuration tuning option to install the Network Driver.
ii. Load the network driver with DDR filters support :

```
[root@host~]# modprobe cxgb4 use_ddr_filters=1
```

iii. Now, create filter rules using cxgbtool:

```
[root@host~]# cxgbtool ethX filter <index> action [pass/drop/switch] fip
<source ip of incoming packet> lip <destination ip of incoming packet> fport
<source port> lport <destination port> hitcnts 1 cap maskless
```

Where,

ethX : Chelsio interface
index *:* Filter index. The user must provide a positive integer, which will be ignored and replaced by an automatically computed index, based on the hash (4-tuple).The index will be displayed after the filter rule is created successfully.
action : Ingress packet disposition
pass : Ingress packets will be passed through set ingress queues
switch : Ingress packets will be routed to an output port with optional header rewrite.
drop : Ingress packets will be dropped

Note *"source IP", "destination IP", "source port" and "destination port" are mandatory parameters since Hash filters don't support masks and hence, 4-tuple must always be supplied for Hash filter. "cap maskless" parameter should be appended in order to create Hash/DDR filter rules. Otherwise the above command will create LE-TCAM filter rules.*

iv. Hash filters will match TCP traffic by default. In order to match other protocol traffic, add *proto* parameter while creating the filter. For example, to match UDP packets:

```
[root@host~]# cxgbtool ethX filter <index> action [pass/drop/switch] fip
<source IP> lip <destination IP> fport <source port> lport <destination
port> proto 17 hitcnts 1 cap maskless
```

3.1.1. Examples

- ***Drop* action**

```
[root@host~]# cxgbtool ethX filter 496 action drop lip 102.1.1.1 fip
102.1.1.2 lport 12865 fport 20000 hitcnts 1 cap maskless iport 1 proto 17
Hash-Filter Index = 61722
```

The above filter rule will drop all UDP packets matching above 4 tuple coming on chelsio port 1.

- ***Pass* action**

```
[root@host~]# cxgbtool ethX filter 496 action pass lip 102.2.2.1 fip
102.2.2.2 lport 12865 fport 12000 hitcnts 1 cap maskless proto 6
Hash-Filter Index = 308184
```

The above filter rule will pass all TCP packets matching above 4 tuple.

- ***Switch* action**

```
[root@host~]# cxgbtool ethX filter 496 action switch lip 102.3.3.1 fip
102.3.3.2 lport 5001 fport 16000 iport 0 eport 1 hitcnts 1 cap maskless
Hash-Filter Index = 489090
```

The above filter rule will switch all the packets matching above 4 tuple from chelsio port 0 to chelsio port 1.

Note *For more information on additional parameters, refer cxgbtool manual by running the `man cxgbtool` command.*

3.2. Listing Filter Rules

- To list the Hash/DDR filters set, run the following command:

```
[root@host~]# cat /sys/kernel/debug/cxgb4/<bus-id>/hash_filters
```

- To list the both LE-TCAM and Hash/DDR filters set, run the following command:

```
[root@host~]# cxgbtool ethX filter show
```

3.3. Removing Filter Rules

To remove a filter, run the following command with *cap maskless* parameter and corresponding filter rule index:

```
[root@host~]# cxgbtool ethX filter index <delete|clear> cap maskless
```

Note

- *Filter rule index can be determined by referring the "hash_filters" file located in /sys/kernel/debug/cxgb4/<bus-id>/.*
- *For more information on additional parameters, refer cxgbtool manual by running the `man cxgbtool` command.*

3.4. Swap MAC feature

Chelsio T5's Swap MAC feature swaps packet source MAC and destination MAC addresses. This is applicable only for switch filter rules. Here's an example:

```
[root@host~]# cxgbtool eth2 filter 1 action switch lip 102.2.2.1 fip
102.2.2.2 lport 5001 fport 14000 hitcnts 1 iport 1 eport 0 swapmac 1 proto
17 cap maskless
Hash-Filter Index = 21936
```

The above example will swap source and destination MAC addresses of UDP packets (matching above 4 tuple) received on adapter port 1 and then switch them to port 0.

Note *This feature is currently supported only with Hash/DDR filters.*

3.5. Hit Counters

For LE-TCAM filters, *hit counters* will work simply by adding *hitcnts 1* parameter to the filter rule. However, for Hash/DDR filters, you will have to make use of tracing feature and RSS queues. Here's a step-by-step guide to enable *hit counters* for Hash/DDR filter rules:

i. Enable tracing on T5 adapter.

```
[root@host~]# cxgbtool ethX reg 0x09800=0x13
```

ii. Setup a trace filter

```
[root@host~]# echo tx1 snaplen=40 > /sys/kernel/debug/cxgb4/<bus_id>/trace0
```

Note *Use "snaplen=60" in case of IPV6.*

iii. Configure RSS Queue to send trace packets. Determine the RspQ ID of the queues by looking at *Trace* QType in */sys/kernel/debug/cxgb4/<bus-id>/sge_qinfo* file

```
[root@host~]# cxgbtool ethX reg 0x0a00c=<Trace Queue0-RspQ ID>
```

The above step will trace all the packets transmitting from port1(tx1) to trace filter 0.

- **Multi-tracing**

To enable *hit counters* for multiple chelsio ports in Tx/Rx direction enable Multi-tracing. Using this we can configure 4 different RSS Queues separately corresponding to 4 trace-filters.

i. Enable Tracing as well as MultiRSSFilter

```
[root@host~]# cxgbtool ethX reg 0x09800=0x33
```

ii. Setup a trace filter

```
[root@host~]# echo tx0 snaplen=40 > /sys/kernel/debug/cxgb4/<bus_id>/trace0
```

iii. Configure the RSS Queue corresponding to trace0 filter configured above. Determine the *RspQ ID* of the queues by looking at *Trace* QType in */sys/kernel/debug/cxgb4/<bus-id>/sge_qinfo* file.

```
[root@host~]# cxgbtool ethX reg 0x09808=<Trace-Queue0-RspQ ID>
```

iv. Similarly for other direction and for multiple ports run the follow commands:

```
[root@host~]# echo rx0 snaplen=40 > /sys/kernel/debug/cxgb4/<bus_id>/trace1
[root@host~]# echo tx1 snaplen=40 > /sys/kernel/debug/cxgb4/<bus_id>/trace2
[root@host~]# echo rx1 snaplen=40 > /sys/kernel/debug/cxgb4/<bus_id>/trace3
[root@host~]# cxgbtool ethX reg 0x09ff4=<Trace-Queue1-RspQ ID>
[root@host~]# cxgbtool ethX reg 0x09ffc=<Trace-Queue2-RspQ ID>
[root@host~]# cxgbtool ethX reg 0x0a004=<Trace-Queue3-RspQ ID>
```

Note *Use “snaplen=60” in case of IPV6.*

XX. Traffic Management

1. Introduction

Traffic Management capabilities built-in to Chelsio adapters can shape transmit data traffic through the use of sophisticated queuing and scheduling algorithms built-in to the ASIC hardware which provides fine-grained software control over latency and bandwidth parameters such as packet rate and byte rate. These features can be used in a variety of data center application environments to solve traffic management problems.

Traffic Management features in Chelsio's adapters allows the user to control three main things:

- Guarantee low latency in the presence of other traffic
- Control max bandwidth that a connection or a flow (a group of connections) can use
- Allocate available bandwidth to several connection or flows based on desired levels of performance

Once the offload transmit traffic shaping classes have been configured, individual offloaded connections (flows) may be assigned to a traffic shaping class in order to manage the flows according to the class configuration. The mechanism to accomplish this "flow to class" mapping assignment is the Connection Offload Policy (COP) configuration system.

1.1. Hardware Requirements

1.1.1. Supported adapters

The following are the currently shipping Chelsio adapters that are compatible with the Traffic Management feature.

- T62100-LP-CR
- T6225-CR
- T520-BT
- T580-CR
- T520-LL-CR
- T520-CR
- T580-LP-CR
- T420-CR
- T440-CR
- T422-CR
- T404-BT
- T420-BCH
- T440-LP-CR
- T420-BT
- T420-LL-CR
- T420-CX

1.2. Software Requirements

1.2.1. Linux Requirements

Currently the Traffic Management feature is available for the following versions:

- RHEL 7.3, 3.10.0-514.el7
- RHEL 7.3, 3.10.0-514.el7.ppc64le (POWER8 LE)
- RHEL 7.3, 4.5.0-15.el7.aarch64 (ARM64)
- RHEL 7.2, 3.10.0-327.el7
- RHEL 6.8, 2.6.32-642.el6
- RHEL 6.7, 2.6.32-573.el6
- SLES 12 SP2, 4.4.21-69-default
- SLES 12 SP1, 3.12.49-11-default
- SLES 11 SP4, 3.0.101-63-default
- Ubuntu 16.04.1, 4.4.0-47-generic
- Ubuntu 14.04.4, 4.2.0-27-generic
- Kernel.org linux-4.8

Other kernel versions have not been tested and are not guaranteed to work.

2. Software/Driver Loading

Important *Please ensure that all inbox drivers are unloaded before proceeding with unified wire drivers.*

Traffic Management can be performed on non-offloaded connections as well as on offloaded connections.

The drivers must be loaded by the root user. Any attempt to load the drivers as a regular user will fail.

Run the following commands to load the TOE driver:

```
[root@host~]# modprobe cxgb4
[root@host~]# modprobe t4_tom
```

3. Software/Driver Unloading

Reboot the system to unload the driver. To unload without rebooting, refer **Unloading the TOE driver** section of **Network (NIC/TOE)** chapter.

4. Software/Driver Configuration and Fine-tuning

4.1. Traffic Management Rules

Traffic Management supports the following types of scheduler hierarchy levels which can be configured using the *cxgbtool* utility:

i. Class Rate Limiting
ii. Class Weighted Round Robin
iii. Channel Rate Limiting

4.1.1. Class Rate Limiting

This scheduler hierarchy level can be used to rate limit individual traffic classes or individual connections (flow) in a traffic class.

Class rate limiting can be configured using the following command:

```
[root@host~]# cxgbtool <ethX> sched-class params type packet level cl-rl
mode <scheduler-mode>  rate-unit <scheduler-rate-unit>  rate-mode
<scheduler-rate-mode> channel <Channel No.> class <scheduler-class-index>
max-rate <maximum-rate>  pkt-size <Packet size>
```

Here,

- *ethX* is the Chelsio interface
- *scheduler-mode* specifies whether the rule is configured for individual traffic classes or individual connections (flow) in a traffic class. Possible values include *flow* or *class*.
- *scheduler-rate-unit* specifies whether the rule is configured for bit-rate or packet rate . Possible values include *bits* or *pkts*
- *scheduler-rate-mode* specifies whether the rule is configured to support a percent of the channel rate or an effective rate. Possible values include *relative* or *absolute.*
- *Channel No.* is the port on which data is flowing (0-3).
- *scheduler-class-index* is the TCP traffic class (0-14 for T4 series of adapters and 0-15 for T5 series of adapters).
- *maximum-rate* is the bit rate (Kbps) for this TCP stream.
- *Packet size* is the TCP mss size in bytes; for example – for an MTU of 1500, use a packet size of 1460.

4.1.2. Class Weighted Round Robin

Incoming traffic flows from various applications can be prioritized and provisioned using a weighted round-robin scheduling algorithm.

Class weighted round robin can be configured using the following command:

```
[root@host~]# cxgbtool <ethX> sched-class params type packet level cl-wrr
channel <Channel No.>  class <scheduler-class-index> weight <Y>
```

Here,

- *ethX* is the Chelsio interface
- *Channel No.* is the port on which data is flowing (0-3).
- *scheduler-class-index* is the TCP traffic class (0-14 for T4 series of adapters and 0-15 for T5 series of adapters).
- *weight* is the weight to be used for a weighted-round-robin scheduling hierarchy. Possible values include 1 to 99.

4.1.3. Channel Rate Limiting

This scheduler hierarchy level can be used to rate limit individual channels.

Channel rate limiting can be configured using the following command:

```
[root@host~]# cxgbtool eth6 sched-class params type packet level ch-rl rate-
unit <scheduler-rate-unit>  rate-mode <scheduler-rate-mode>  channel 1 max-
rate <maximum-rate>
```

Here,

- *ethX* is the Chelsio interface
- *scheduler-rate-unit* specifies whether the traffic management rule is configured for bit-rate or packet rate . Possible values include *bits* or *pkts*
- *scheduler-rate-mode* specifies whether the traffic management rule is configured to support a percent of the channel rate or an effective rate. Possible values include *relative* or *absolute.*
- *Channel No.* is the port on which data is flowing (0-3).
- *maximum-rate* is the bit rate (Kbps) for this TCP stream. The lower limit is 1 Gbps.

4.2. Configuring Traffic Management

4.2.1. For Non-offloaded connections

Traffic Management of non-offloaded connections is a 2-step process. In the first step bind connections to indicated NIC TX queue using *tc* utility from *iproute2-3.9.0* package. In the second step bind the indicated NIC TX queue to the specified TC Scheduler class using the *cxgbtool* utility.

i. Load the network driver and bring up the interface:

```
[root@host~]# modprobe cxgb4
[root@host~]# ifconfig ethX up
```

ii. Bind connections to queues:

```
[root@host~]# tc qdisc add dev ethX root handle 1: multiq
[root@host~]# tc filter add dev ethX parent 1: protocol ip prio 1 u32  match
ip dst <IP address of destination> action skbedit queue_mapping <queue>
```

Note *For additional binding options, run [root@host~]# man tc*

iii. Now, bind the NIC TX queue with traffic class:

```
[root@host~]# cxgbtool ethX sched-queue <queue> <class>
```

Here,

- *ethX* is the Chelsio interface
- *queue* is the NIC TX queue
- *class* is the TX scheduler class

Note *If the TX queue is all, * or any negative value, the binding will apply to all of the TX queues associated with the interface. If the class is unbind, clear or any negative value, the TX queue(s) will be unbound from any current TX Scheduler Class binding.*

4.2.2. For Offloaded connections

Traffic Management of offloaded connections can be configured either by applying *COP* policies that associate offloaded connections to classes or by modifying the application.

Both the methods have been described below:

- **Applying *COP* policy**

i. Load the TOE driver and bring up the interface:

```
[root@host~]# modprobe t4_tom
[root@host~]# ifconfig ethX up
```

ii. Create a new policy file (say *new_policy_file*) and add the following line to associate connections with the given scheduling class.

E.g.:

src host 102.1.1.1 => offload class 0

The above example will associate all connections originating from IP address 102.1.1.1 with scheduling class 0

Note *If no specified rule matches a connection, a default setting will be used which disables offload for that connection. That is, there will always be a final implicit rule following all the rules in the input rule set of:*

all => !offload

iii. Compile the policy file using *COP*

```
[root@host~]# cop -d –o <output_policy_file> <new_policy_file>
```

iv. Apply the *COP* policy:

```
[root@host~]# cxgbtool ethX policy <output_policy_file>
```

Where,

ethX: Chelsio interface

Note *For more information on additional parameters, refer cop manual by running the* `man cop` *command.*

- **Modifying the application**

The application can also be modified in order to associate connections to scheduling classes. Follow the steps mentioned below:

i. Determine the TCP socket file descriptor in the application through which data is sent.
ii. Declare and initialize a variable in the application:

```
int cl=1;
```

Here,

- *cl* is the TCP traffic class(scheduler-class-index) that the user wishes to assign the data stream to. This value needs to be in the range of 0 to 7.

The application will function according to the parameters set for that traffic class.

iii. Add socket option definitions:

In order to use *setsockopt()* to set the options to the TCP socket, the following two definitions need to be made:

- SOL_SCHEDCLASS used for setting TCP traffic class, which has the value 290.
- IPPROTO_TCP used for setting the type of IP Protocol.

```
# define SOL_SCHEDCLASS 290
# define IPPROTO_TCP 6
```

iv. Use the *setsockopt()* function to set socket options:

The *setsockopt()* call must be mentioned after the connect() call.

```
//Get the TCP socket descriptor variable
setsockopt (sockfd , IPPROTO_TCP, SOL_SCHEDCLASS, &cl, sizeof(cl));
```

Here:

- *sockfd* : The file descriptor of the TCP socket
- *&cl* : Pointer to the class variables
- *sizeof(cl)* : The size of the variable

v. Now, compile the application.

5. Usage

5.1. Non-Offloaded Connections

The following example demonstrates the method to rate limit all TCP connections on class 0 to a rate of 300 Mbps for Non-offload connections:

i. Load the network driver and bring up the interface:

```
[root@host~]# modprobe cxgb4
[root@host~]# ifconfig eth0 up
```

ii. Bind connections with destination IP address 192.168.5.3 to NIC TX queue 3

```
[root@host~]# tc qdisc add dev eth0 root handle 1: multiq
[root@host~]# tc filter add dev eth0 parent 1: protocol ip prio 1 u32  match
ip dst 192.168.5.3 action skbedit queue_mapping 3
```

iii. Bind the NIC TX queue to class 0

```
[root@host~]# cxgbtool eth0 sched-queue 3 0
```

iv. Set the appropriate rule for class 0

```
[root@host~]# cxgbtool eth0 sched-class params  type packet  level cl-rl
mode class  rate-unit bits  rate-mode absolute channel 0  class 0 max-rate
300000  pkt-size 1460
```

5.2. Offloaded Connections

The following example demonstrates the method to rate limit all TCP connections on class 0 to a rate of 300 Mbps for offloaded connections:

i. Load the TOE driver and bring up the interface

```
[root@host~]# modprobe t4_tom
[root@host~]# ifconfig eth0 up
```

ii. Create a new policy file (say *new_policy_file*) and add the following line to associate connections with the given scheduling class.:

```
src host 102.1.1.1 => offload class 0
```

iii. Compile the policy file using *COP*

```
[root@host~]# cop -d -o <output_policy_file> <new_policy_file>
```

iv. Apply the *COP* policy:

```
[root@host~]# cxgbtool eth0 policy <output_policy_file>
```

v. Set the appropriate rule for class 0

```
[root@host~]# cxgbtool ethX sched-class params  type packet  level cl-rl
mode class  rate-unit bits  rate-mode absolute channel 0  class 0 max-rate
300000  pkt-size 1460
```

5.3. Offloaded Connections with Modified Application

The following example demonstrates the method to rate limit all TCP connections on class 0 to a rate of 300 Mbps for for offloaded connections with modified application.

i. Load the TOE driver and bring up the interface.

```
[root@host~]# modprobe t4_tom
[root@host~]# ifconfig eth0 up
```

ii. Modify the application as mentioned in the Configuring Traffic Management section.

iii. Set the appropriate rule for class 0

```
[root@host~]# cxgbtool ethX sched-class params  type packet  level cl-rl
mode class  rate-unit bits  rate-mode absolute channel 0  class 0 max-rate
300000  pkt-size 1460
```

XXI. Unified Wire Manager (UM)

1. Introduction

Chelsio's Unified Wire Manager is a powerful management software tool, allowing you to view and configure different aspects of the system, including Chelsio hardware installed in the system. The software includes a command line interface (CLI) tool and a web management interface (Web GUI) to help you manage all Chelsio network adapters on the network across multiple operating systems.

Unified Wire Manager enables the management of all aspects of the client side of the iSCSI SAN in two main areas. The ability to configure Chelsio adapter's boot option ROM without entering each individual adapter's configuration screen and manage group of iSCSI initiators remotely from a common user interface saves administrator's time considerably. Unified Wire Manager fully supports Linux's Open-iSCSI initiator and Chelsio's iSCSI Target. All supported Chelsio FCoE initiators available on Linux and Windows operating systems can be managed.

Users can manage Option ROM (PXE and iSCSI for T3; PXE, FCoE and iSCSI for T5 and T4) capability for Chelsio cards using various tools available in the software.

Additionally, Unified Wire Manager allows Chelsio adapter NIC and TOE parameters to be centrally managed through the same easy-to-use user interface. It can bring interfaces up or down, team/bond interfaces together, tune parameters for optimal performance, and any number of administrative tasks normally done at each individual machine.

Chelsio's Unified Wire Manager is an indispensable tool for saving administrator's time for managing Network and SAN resources. Chelsio's high performance network adapters with its Unified Wire approach to networking can now be managed centrally in a simple and fast way!

1.1. Features

Chelsio's Unified Wire Manager is designed to provide the following features to the end-user:

- Remotely manage Chelsio adapters and various related tasks like driver installation from a single application.
- Single tool with CLI and Web interface that works across Storage, Networking and Hardware.
- Manage all Chelsio adapters installed on the system.
- Tool for FAE to debug issues on the Customer front.
- Freedom to choose various modes of management i.e. CLI or Web GUI.

1.2. Reference Architecture

Chelsio's Web GUI is a web-based management interface that lets you remotely manage several Chelsio CNAs from anywhere, at anytime on the network using a web browser. The Web GUI provides a great amount of flexibility, efficiency and accessibility to system administrators in managing Network and SAN resources. The users have the freedom to access the interface using any of the major browsers available, based on individual preferences and corporate policy.

System performance degradation issues will not be observed when using the Web GUI, since it's lightweight and utilizes very less system resources.

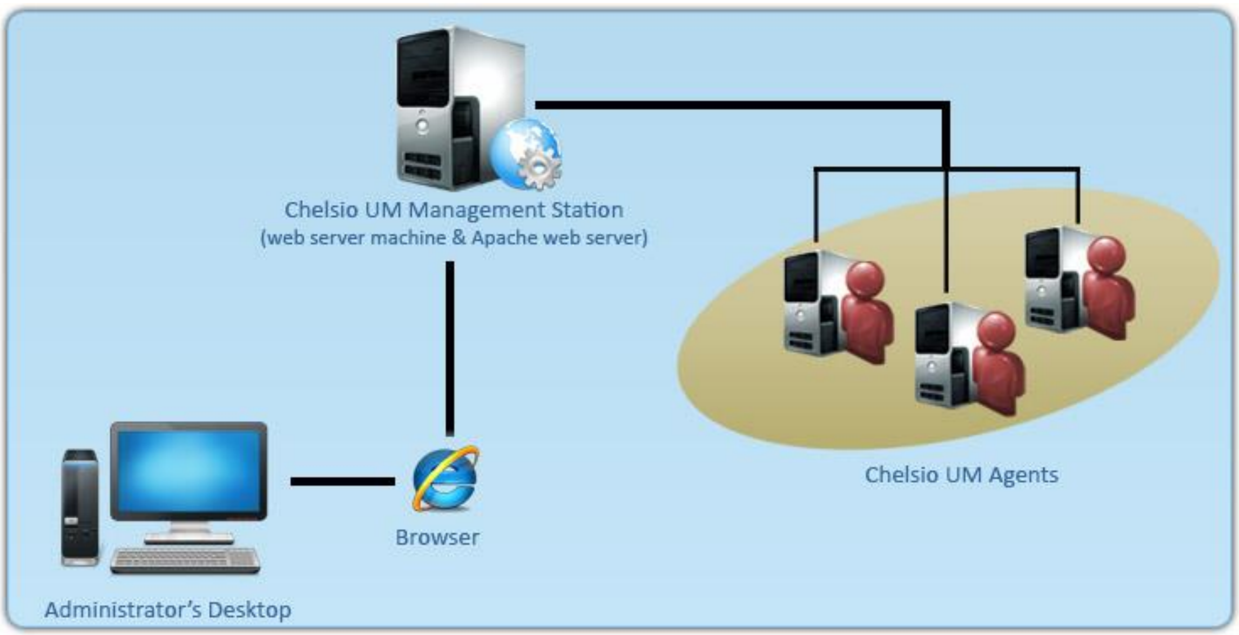


Figure 1.2 – Chelsio Unified Wire Manager with web interface (Web GUI)

1.3. Unified Wire Manager Components

1.3.1. Management Agent

The Management Agent is a binary executable, which runs as a service on the system that has at least one Chelsio card installed. It is installed along with libraries that can manage various components of the system and enabled during system startup.

1.3.2. Management Client

The Management Client can be used to connect to agents and manage them. Once connected you can view and configure Chelsio HBAs and related networking, storage and hardware properties. You can use either the CLI or Web GUI client to manage agents based on your preference. It makes service requests based on the command issued by the user and returns the appropriate information.

- ### CLI Client

The **CLI Client** (*chelsio_uwcli*) is an executable binary which allows you to manage and configure agents using the command-line interface. It is not a command shell with a prompt; it accepts all command parameters as arguments when launching it, making it script-friendly.

- ### Web GUI Client

The **Web Management Interface** (Web GUI) client is a web-based management interface which allows you to securely manage agents from anywhere using a web browser. The management interface uses a secure 256-bit encrypted HTTP connection, ensuring that authentication and configuration data are protected during transmission from the web browser to the system and vice versa. Many agents can be accessed on single interface making it very efficient & user-friendly.

Currently supported browsers are **Internet Explorer 9+**, **Mozilla Firefox 3.6.9+**, **Google Chrome 5+** and **Apple Safari 5+**.

1.4. Authentication and encryption

The Unified Wire Manager requires user authentication to manage a system. A user must have administrative privileges to manage a system. The authentication credentials, as well as all data exchanged between the CLI client or the Web GUI and the agent, are encrypted using SSL. This ensures that the data cannot be accessed when it is being transmitted over the network.

2. Hardware and Software

2.1. Supported adapters

Following are the currently shipping Chelsio adapters that are compatible with Chelsio Unified Wire Manager:

- T62100-LP-CR
- T6225-CR
- T520-BT
- T580-CR
- T520-LL-CR
- T520-CR
- T580-LP-CR
- T420-CR
- T440-CR
- T422-CR
- T420-SO-CR
- T404-BT
- T420-BCH
- T440-LP-CR
- T420-BT
- T420-LL-CR
- T420-CX
- S302E
- S302E-C
- S310E-CR
- S310E-CR-C
- S310E-CXA
- S310E-SR+
- S310E-SR
- S310E-BT
- S320E-CR
- S320E-LP-CR
- S320E-CXA
- S320EM-BS
- S320EM-BCH
- N320E-G2-CR
- N320E
- N320E-CXA
- N320E-BT
- N310E
- N310E-CXA

2.2. Platform/Component Matrix

The table below lists the Linux distributions and the supported UM components.

Distribution	Supported UM Components
RHEL 6.6, 2.6.32-504.el6	Management Agent, Management Client, Web Management Interface
SLES 11 SP3, 3.0.76-0.11-default	Management Agent, Management Client, Web Management Interface

2.3. Platform/Driver Matrix

The table below lists the Chelsio drivers and their supported versions:

Chelsio driver	Version
NIC	T3: 2.0.0.1(RHEL 5.8,6.3; inbox driver for SLES11SP3) T4,T5: 3.1.0.0
TOE	T3 : 2.0.0.1 (RHEL 5.8,6.3) T4,T5: 3.1.0.0
Bypass	T4: 3.1.0.0
Bonding	T3 : 2.0.0.1 T4: 3.7.1
iSCSI Target	T3: 5.2.0-0560 T4,T5: 3.1.0.0-1241
Open iSCSI Initiator	T3,T4,T5: 2.0-873
iWARP	T3,T4,T5: 3.1.0.0
FCoE Initiator	T4: 3.1.0.0

3. Installing Unified Wire Manager

Chelsio Unified Wire has been designed to install Unified Wire Manager (UM) by default. All the three UM components, i.e. Management Agent, Client and Station, will be installed on selecting any of the Terminator 4/Terminator 5 configuration tuning options during installation. Hence, no separate installation is required.

4. Verifying UM components status

The following section explains how to verify status of various UM components.

4.1. Verifying Management Agent

i. Execute the following query command :

```
[root@chelsio]# ps -eaf | grep UW
```

The above query should confirm that Management Agent is running by displaying a similar result:

```
root      30531      1  0 09:27 ?       00:00:00 ./UWMgrServer
root      30534      1  0 09:27 ?       00:00:00 ./UWMgrServer --run=slp
root      30537      1  0 09:27 ?       00:00:00 ./UWMgrServer --run=logserver
root      30581  28384  0 09:45 pts/1   00:00:00 grep UW
```

ii. You can also execute the following comand to determine if Management Agent is running:

```
[root@chelsio]# /etc/init.d/chelsio-uwire_mgmtd  status
```

The above command should display the following result:

```
Unified Wire Manager Agent : Running
```

4.2. Verifying Management Client

Execute the following query command to determine if Management Client is installed:

```
[root@host~]# chelsio_uwcli -V
```

The above query should confirm that Management Client is installed by displaying a similar result:

```
Unified Manager client CLI version : 2.x.yy
```

4.3. Verifying Management Station

Execute the following query command to determine the status of Management Station:

```
[root@host~]# /etc/init.d/chelsio-mgmtstd status
```

The above command will display one of the following messages:

UM Management Station: Running
UM Management Station: Stopped

Based on the status displayed, you can start, stop or restart Management Station by executing the following command:

```
[root@host~]# /etc/init.d/chelsio-mgmtstd [start|stop|restart]
```

Note *While accessing the Web Management Interface, if an error "Management Station is not running" is displayed at the log-in page, follow the above steps to restart Management Station and try again.*

5. Management Agent

5.1. Communication

The agent uses a TCP connection over IP to communicate with the client. After the connection is established, SSL (Secure Sockets Layer) encryption is enabled using the Open SSL libraries. The agent listens on a TCP port for new incoming connections from clients. This port is set to 35001 by default. It may be changed in the configuration file for the agent. The agent needs to be restarted after the change.

5.2. Configuration

The agent uses a configuration file *uwmgr.conf*, which is in the agent's installation directory in */etc/chelsio-uwire*. The only configurable parameter available for the agent is the TCP listening port, which can be specified with syntax similar to the example below: **PORT 35001**. After changing the port, please restart Management Agent for changes to take effect.

5.3. Service configuration

The agent is installed as a service on the system, and enabled to start on boot. The following sections will describe the procedure to configure service startup manually:

5.3.1. Service startup configuration

The service name is *chelsio-uwire_mgmtd* and can be configured using the *chkconfig* utility. The service startup configuration can be viewed and modified as below:

Execute the following command to list the service configuration.

```
[root@host~]# chkconfig --list chelsio-uwire_mgmtd
```

Execute the following command to enable/disable the service to start at system runlevel 5.

```
[root@host~]# chkconfig --level 5 chelsio-uwire_mgmtd on/off
```

5.3.2. Service start/stop/restart

You can start, stop or restart the service by using the following command:

```
[root@host~]#/etc/init.d/chelsio-uwire_mgmtd [start|stop|restart]
```

5.4. Firewall

If the system has a firewall configured, such as *iptables*, it should be configured to allow traffic to the management agent TCP port configured above in the configuration section, or the default port that the management agent uses, 35001. Review the firewall documentation and configure it appropriately. If there is a firewall appliance / software protecting the network that the system is on, and you wish to connect to the system from a different network, using the client, the firewall appliance also needs to be configured appropriately.

6. CLI client

6.1. CLI Help system

A detailed help and usage documentation is built into the CLI, and is accessible through its help system. The help can be invoked by the usual argument of /? or --help.

6.1.1. Viewing help

Use the *chelsio_uwcli* command to view the help file as shown below:

```
[root@host~]# chelsio_uwcli /?
```

6.2. Client conflict resolution

The CLI and Web GUI cannot manage the same system at the same time by default. This is to ensure that configuration changes being applied by one client are not interrupted by another client. Also, two different Web GUI or CLI clients cannot connect to a management agent at the same time. There is no mechanism to allow this scenario.

7. Web GUI client

7.1. Management Station

In order to access the Web Management Interface, Apache HTTP server should be installed and running on a machine. Also, Cookies and Javascript must be enabled in the browser.

7.1.1. Running Management Station on RHEL 6.x

i. Start/Restart Apache httpd daemon:

```
[root@host~]# service httpd [start|restart]
```

ii. Start/Restart the Management Station:

```
[root@host~]# /etc/init.d/chelsio-mgmtstd [start|restart]
```

7.1.2. Running Management Station on SLES11SP3

i. On SLES11SP3, Management Station needs to be configured before running. Hence, execute the following in command prompt and provide valid inputs.

```
[root@host~]# cd /etc/apache2
[root@host~]# openssl genrsa -des3 -out server.key 1024
[root@host~]# openssl req -new -key server.key -out server.csr
[root@host~]# cp server.key server.key.org
[root@host~]# openssl rsa -in server.key.org -out server.key
[root@host~]# openssl x509 -req -days 365 -in server.csr -signkey server.key
-out server.crt
[root@host~]# cp server.crt ./ssl.crt
[root@host~]# cp server.key ./ssl.key
```

ii. Start/Restart Apache services

```
[root@host~]# rcapache2 [start|restart]
```

iii. Start/Restart the Management Station:

```
[root@host~]# /etc/init.d/chelsio-mgmtstd [start/restart]
```

7.2. Accessing Web Management Interface

i. To access the Web GUI, type in the URL https://<management station IP address> in a web browser.
ii. The security certificate used by the web server is a generic one. It may cause the following types of prompts in different browsers. You will need to select the correct option to continue.

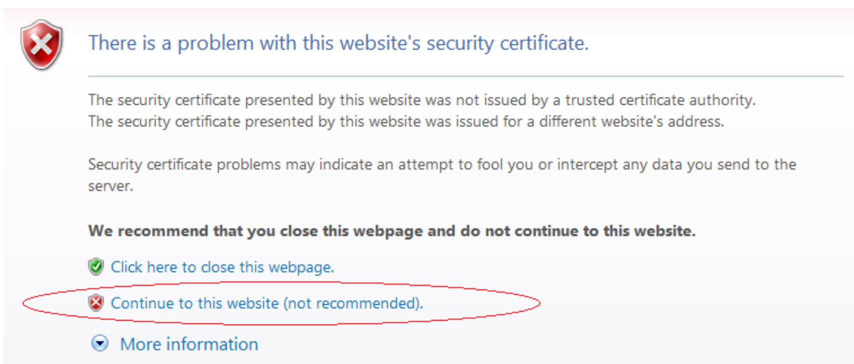
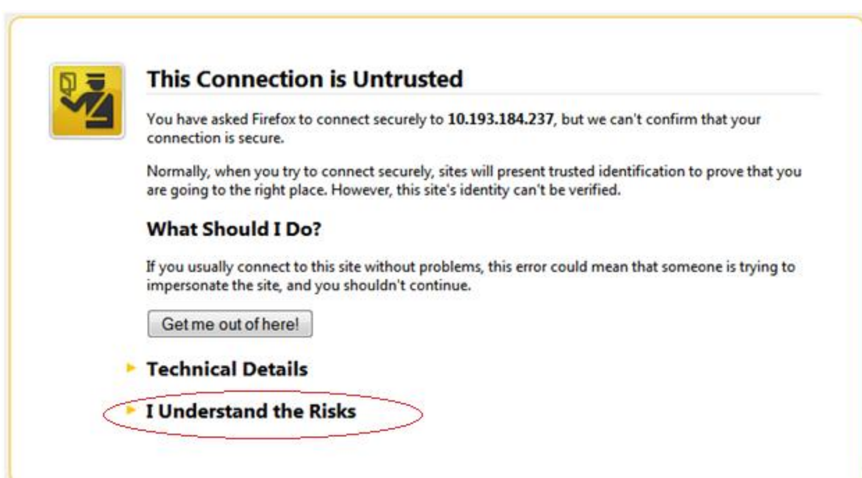


Figure 7.2 (a) - Security Certificate prompt in Internet Explorer



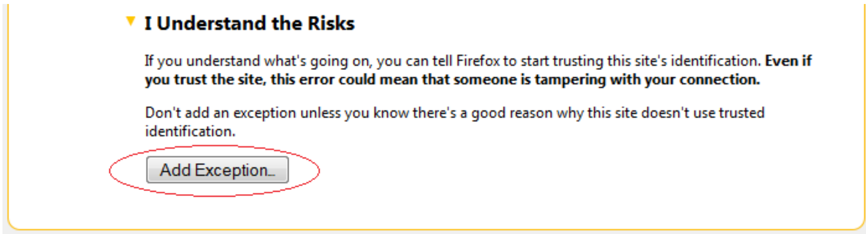


Figure 7.2 (b) - Security Certificate prompt in Mozilla Firefox

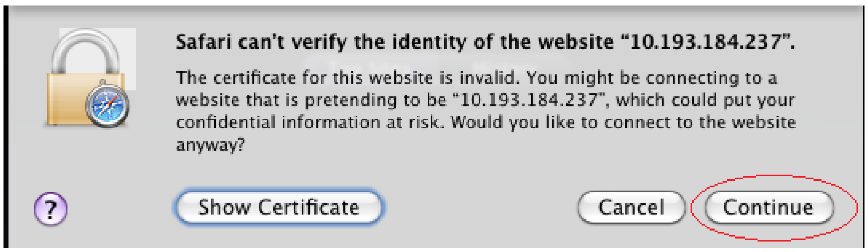


Figure 7.2 (c) - Security Certificate prompt in Apple Safari

Figure 7.2 (d) - Security Certificate prompt in Google Chrome

iii. The web interface requires password authorization to be accessed. Enter the administrator/root credentials that were set up on the management station system and click on the **Login** button.

Figure 7.2 (e) - Web GUI Login page

Note *Not performing any operation/action for 5 minutes will result in session timeout. You will have to re-login and connect to the Agents again.*

7.3. Layout and Navigation

The Web Management Interface consists of the following:

- **Title bar** displaying the username on the left, Unified Wire Manager logo and name in the centre; and a Logout button on the right.
- **Menu Bar** consisting of the **Home, Add System, Remove System, Refresh, Subscribe** and **Bulk Configuration** buttons**.**
- The **Navigation Pane** with a cascading tree of links to various configuration modules for a UM Agent. You can navigate between connected agents and various sections of the managed agent's interface. You can view and hide the configuration modules for each Agent by clicking on the "**+**"and "**-**"links respectively
- The **Details Pane** on the right displaying panels associated with the tree menu item selected in the **Navigation Pane.** The panels can be expanded and collapsed by clicking on the panel heading.
- The **Bottom bar** has the **About** link on the right and copyright details on the left.

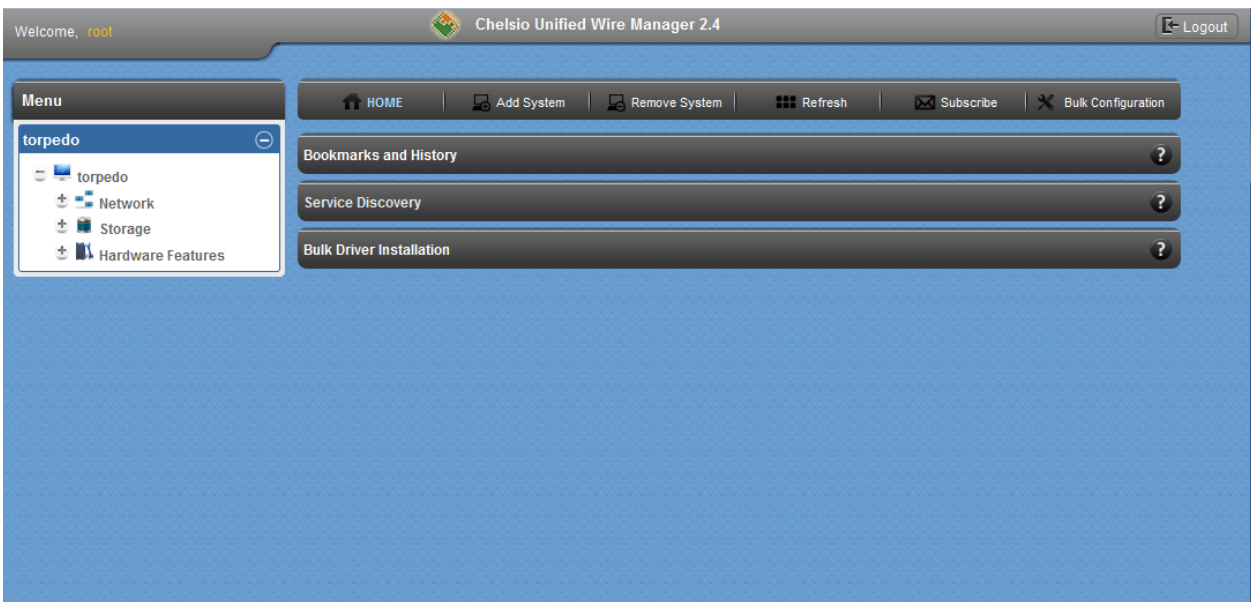


Figure 7.3 – Web Management Interface

7.4. Home page

The home page is displayed by default on launching the Web GUI. It displays **Bookmarks and History**, **Service Discovery** and **Bulk Driver Installation** modules. Options to go back to home page, add/remove system, refresh and configure email alerts are also available.

7.4.1. Home

This option will display the home page.

- **Bookmarks and History**

A history of the last 128 systems that were managed from this system, by the current user, will be shown here in a list. Each system's management IP address, TCP port, and Login details are also stored. This may be edited and saved. Any systems that are not required in the list may be deleted.

Important ***Storing login passwords for the managed systems is inherently insecure****. The passwords are encrypted, but it is still advisable to store passwords only if the system you are running the GUI client on, is secure.*

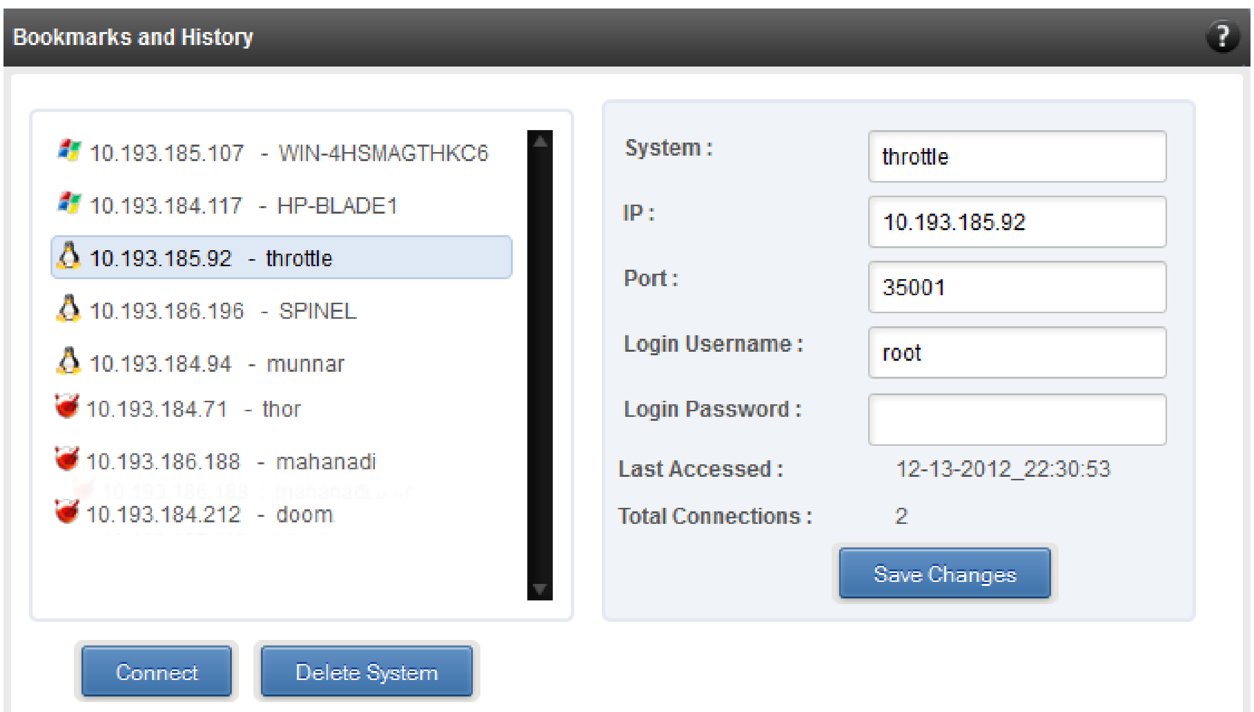


Figure 7.4.1 (a) - Bookmarks and history module

- **Connecting to a system**

Select the system from the Bookmark list and click **Connect**. Once successfully connected, the system will appear on the left pane with different related modules on the right to view and manage.

- **Removing a system**

Select the system from the Bookmark list and click *Delete system* to remove it.

Note *Once removed, the system will no longer appear in the Bookmarks and History module. If you wish to manage that system again, you will have to use the* ***Add system*** *option.*

Service Discovery

Using this module, all the Unified Wire Manager agents connected in the same or different subnet can be discovered. One can choose to discover agents based on OS type or search for a particular agent if the agent's IP or hostname is known. Select the appropriate discovery method and provide the relevant information. For example, to search using hostname, select *Hostname* as the **Input Type** and provide the agent's hostname in the **Search for Hostname/IP** field. Finally click **Discover Agents**.

The **Add Agents** button adds the selected system to the list of discovered agents in the **Bookmarks and History** module. The **Clear Agents** button resets the list of discovered agents.

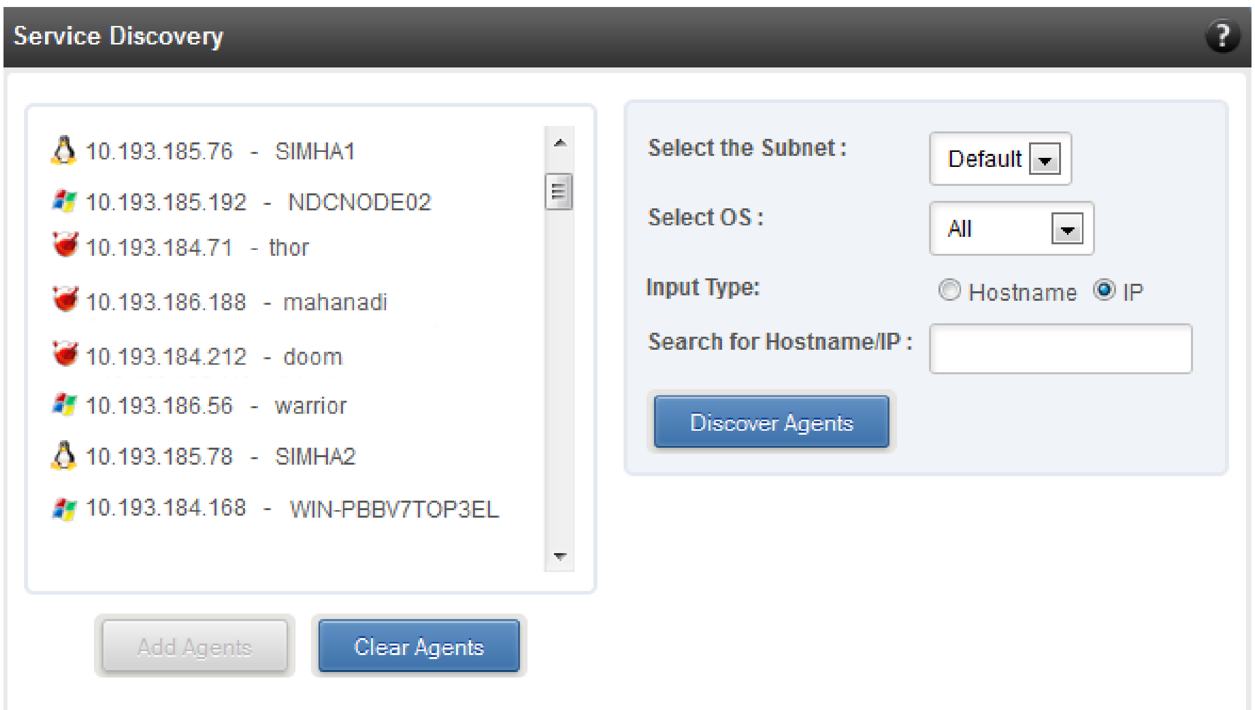


Figure 7.4.1 (b) - Service Discovery module

Bulk Driver Installation

This module allows you to install drivers for multiple systems simultaneously. Drivers available for installation for a particular system may differ depending on the network adapter (T5, T4 or T3) and operating system selected.

- **Installing Driver**

1. In the **Choose the card** fields, select T3 or T4/T5 depending on the chip revision of the network card.
2. Select the operating system for which drivers are to be installed in the **Choose the OS Type** field. All the systems with the selected operating system will be displayed in the list below.
3. Select a system or systems from the list and choose the driver to be installed in the **Driver Installation** section.
4. Download the appropriate driver from Chelsio's Download Center, service.chelsio.com.
5. Locate the driver package.
6. Click **Install** button to install the driver.

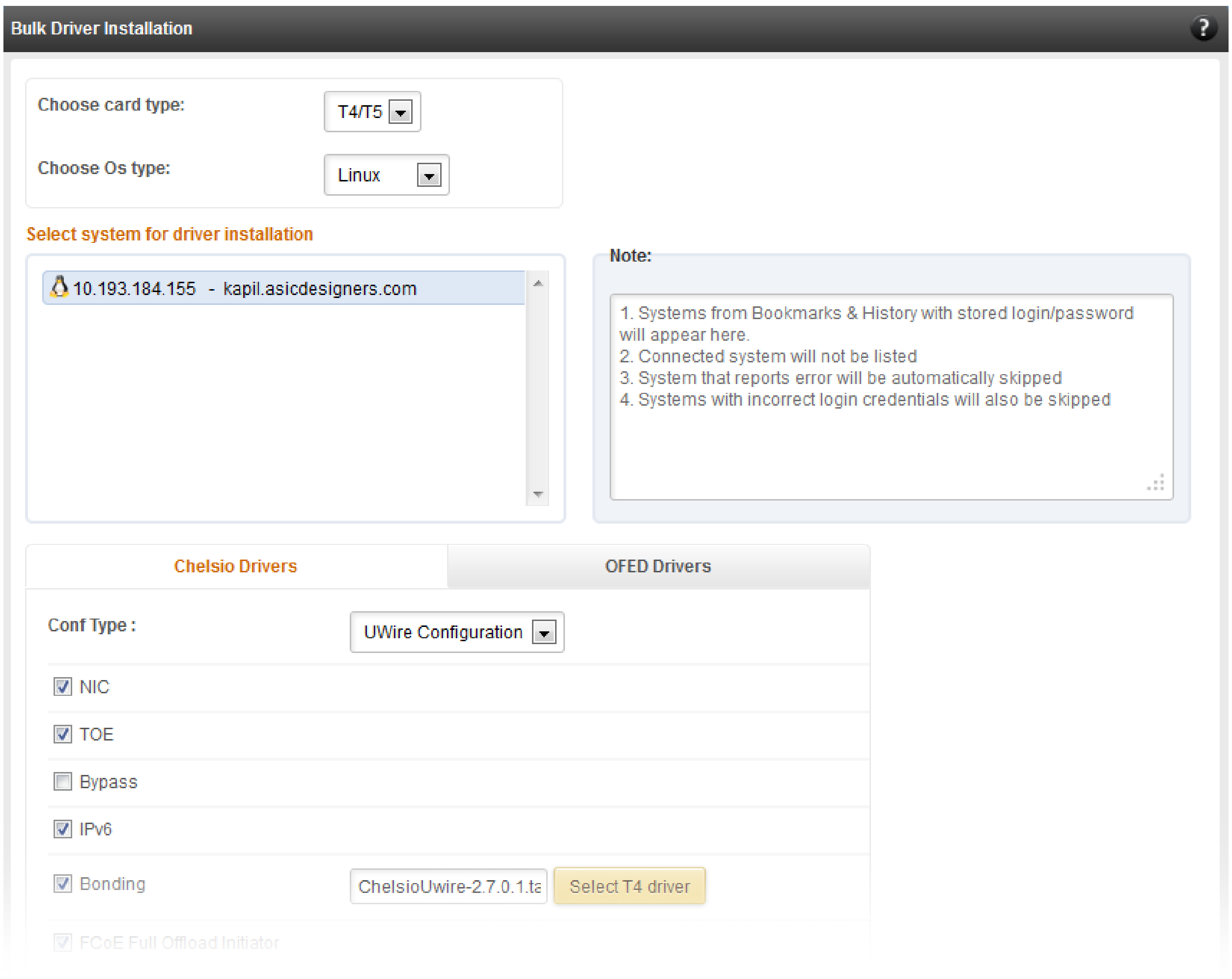


Figure 7.4.1 (c) – Bulk Driver Installation module

Note *Agents that report errors or with incorrect login credentials will be automatically skipped during the driver installation.*

7.4.2. Add System

Use this option to connect to new Agents using their IP or Hostname. You can enter the TCP port for connection or leave it at its default value (35001). You will have to provide correct user credentials for the agent in order to connect successfully.

After connecting to the Agent, the menu bar on the left will display the connected system and its related modules.

If you deselect the 'Remember Password' option, you will be asked to enter password every time you try to connect to the system.

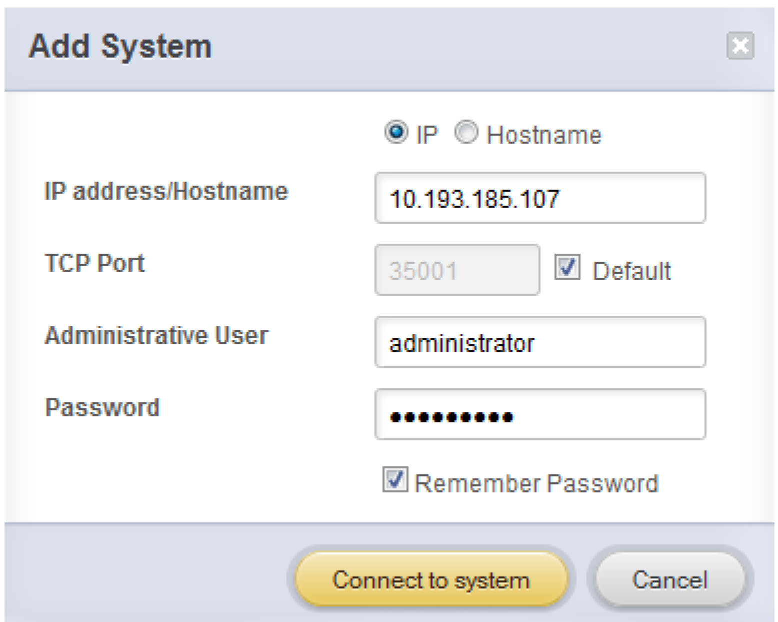


Figure 7.4.2 - Adding a UM Agent

7.4.3. Remove System

Use this option to disconnect an Agent. To remove an agent, click on the name of the system in the tree menu in the left and click **Remove System**. Then click **Yes** to confirm.

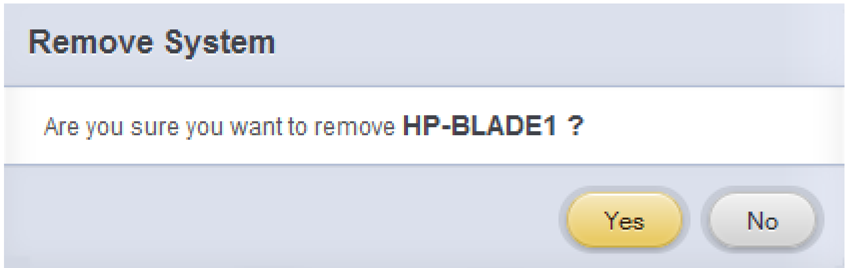


Figure 7.4.3 - Removing a UM Agent

7.4.4. Refresh

This option can be used to reload the Web GUI or UM Agent.

To reload the Web GUI, navigate to the Home page (by clicking on the “Home” button and click Refresh. You can use this option to refresh Home page panels (**Bookmarks and History**, **Service Discovery** and **Bulk Driver Installation**).

To reload an Agent, click on the name of the system in the tree menu in the left and click “Refresh”. You can use this option to update any changes made to system settings like load/unload drivers.

7.4.5. Subscribe (Email Alerts)

This feature is available only on the Web Management Interface.

Using this option, you can receive email alerts regarding the link status of a Chelsio Network Interface Card. This feature sends email notifications regarding the port and the card, on which the link up/down event is occurring. Not only can you configure multiple email addresses to receive notifications, but also customize the email id of the sender for troubleshooting purposes.

To subscribe to **Email Alerts,** enter the sender’s email address in the *Email address* field. It can be anything in the format of <name>@<domain>.<extension>. You can enter multiple email addresses for the *Recipients* field separated a comma. Enter Mail server details and ensure that the “Enable email Alerts” field is enabled. Select the Agent(s), for which you want to receive alerts and Click on **Save.**

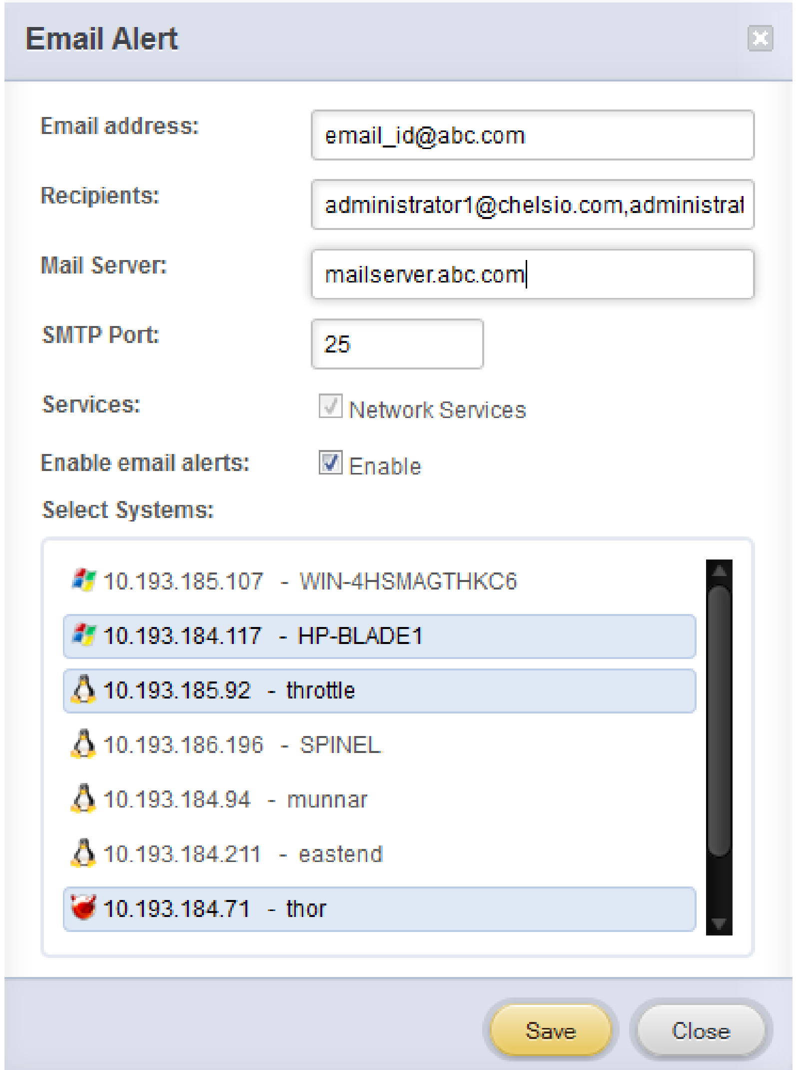


Figure 7.4.5 - Subscribing to Email Alerts

7.4.6. Bulk Configuration

The **Bulk Configuration** page allows you to execute common configuration changes to multiple agents and their network adapters simultaneously. You can conveniently perform bulk operations like installing option ROM, setting MTU and VLAN ID, changing adapter and port parameters on various devices, without having to access multiple modules and thus saving considerable amount of administration time.

Various configurable parameters have been categorized into several modules like **Boot Configuration** module to install and erase option ROM, **Network Configuration** module to set MTU and VLAN ID, **Card Configuration** module to change driver parameters, etc.

Before accessing these modules, you will have to create ***groups*** and then add ***members*** to that group. Once done, you can select the group in the modules and the new setting will be applied to all members of that particular group.

Manage Groups

This is where you can add, delete and manage groups. Use the **Create a Group** section to create a group by specifying agent's platform and group type. There are various types of groups to choose from depending on the type of configuration setting you want to change. For example, to change the MTU size of a network interface (in the Network Configuration module), create a group with group type *Network*. To install or erase option ROM on a Chelsio T4 adapter (in the Boot Configuration module), create a group with group type *t4adapter*.

Here is a list of available configuration modules and corresponding group type:

- **Boot Configuration:** t3adapter,t4adapter,t5adapter
- **Network Configuration:** Network
- **Card Configuration:** t3adapter,t4adapter,t5adapter
- **Port Configuration:** t3port, t4port,t5port
- **Bypass Configuration:** Network

After the group has been created, add members to that group using the **Add a member row** button. Depending on the group type selected, you may be asked to provide additional details for the new member. Use the **Fetch Details** button to do so and finally click **Save a member** button to add the member to the group.

To delete a group, select it from the **Select a Group** drop-down list, and click **Delete Group.** To delete a member from a group, select the group to which the member belongs, select the radio button corresponding to the member to be deleted in the **SNO** field and finally click **Delete Member(s).**

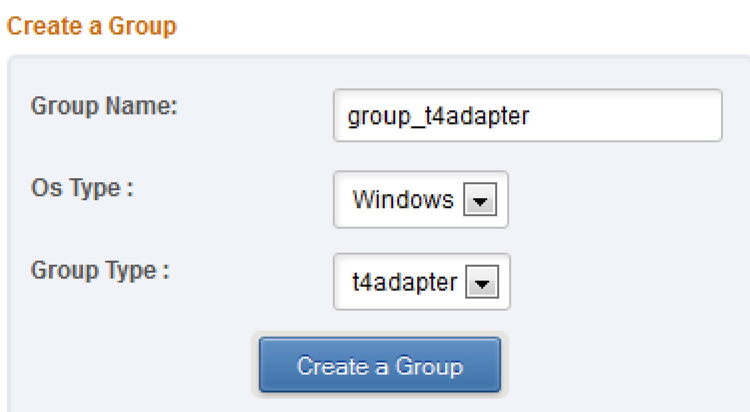


Figure 7.4.6 (a) – Creating a group

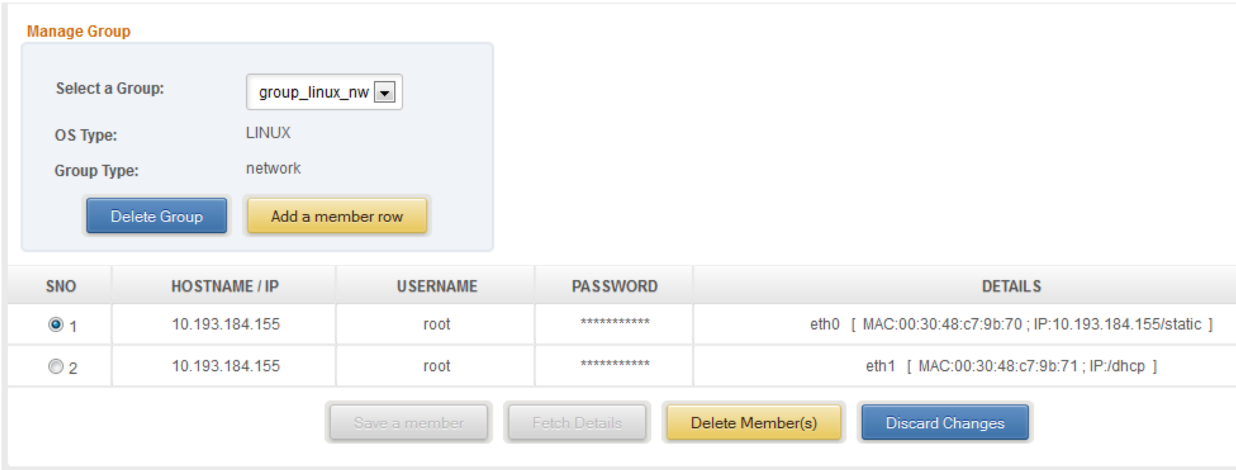


Figure 7.4.6 (b) - Managing a group

Boot Configuration

Using this module, you can install option ROM or erase option ROM on Chelsio network devices. The **Set Default Boot Settings** button will reset the adapter to factory boot settings.

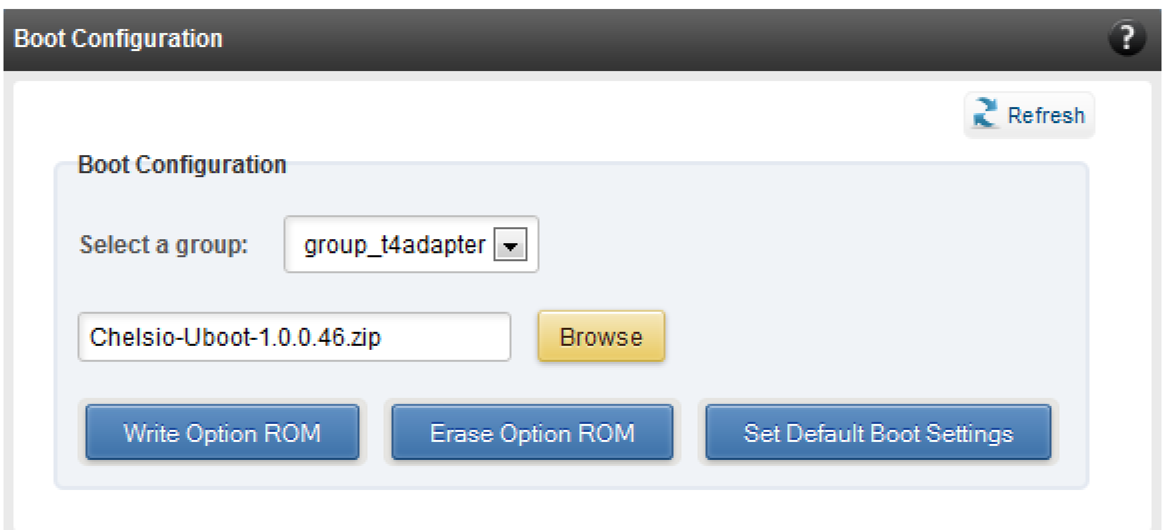


Figure 7.4.6 (c) – Boot Configuration module

Network Configuration

In the **Network Configuration** module, you can set Maximum Transfer Unit (MTU),Virtual LAN (VLAN) ID and change the IP address type for the members (network interfaces) of the *Network* group. MTU can be set between 1500-9000 bytes. VLAN id can be set for an adapter within the range 0-4094 (enter 0 to disable it).

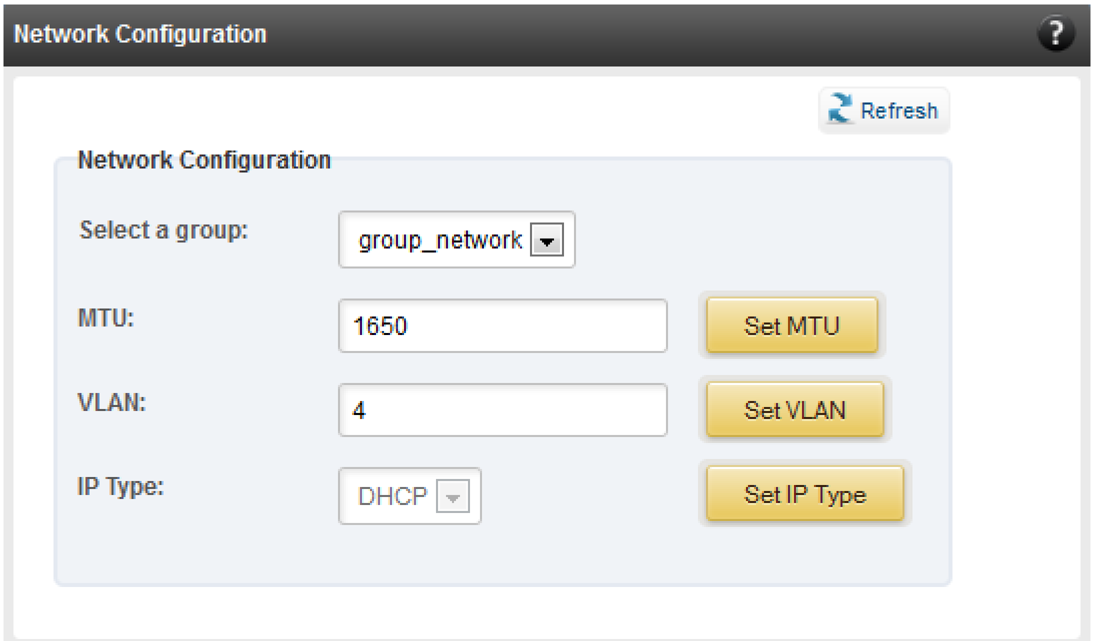


Figure 7.4.6 (d) – Network Configuration module

Card Configuration

The **Card Configuration** module allows you to set various adapter settings including TCP Offload. Offload settings are only available when using the TOE capable drivers (*t4_tom* and *toecore* for T5 and T4 adapters; *t3_tom* and *toecore* for T3 adapters).

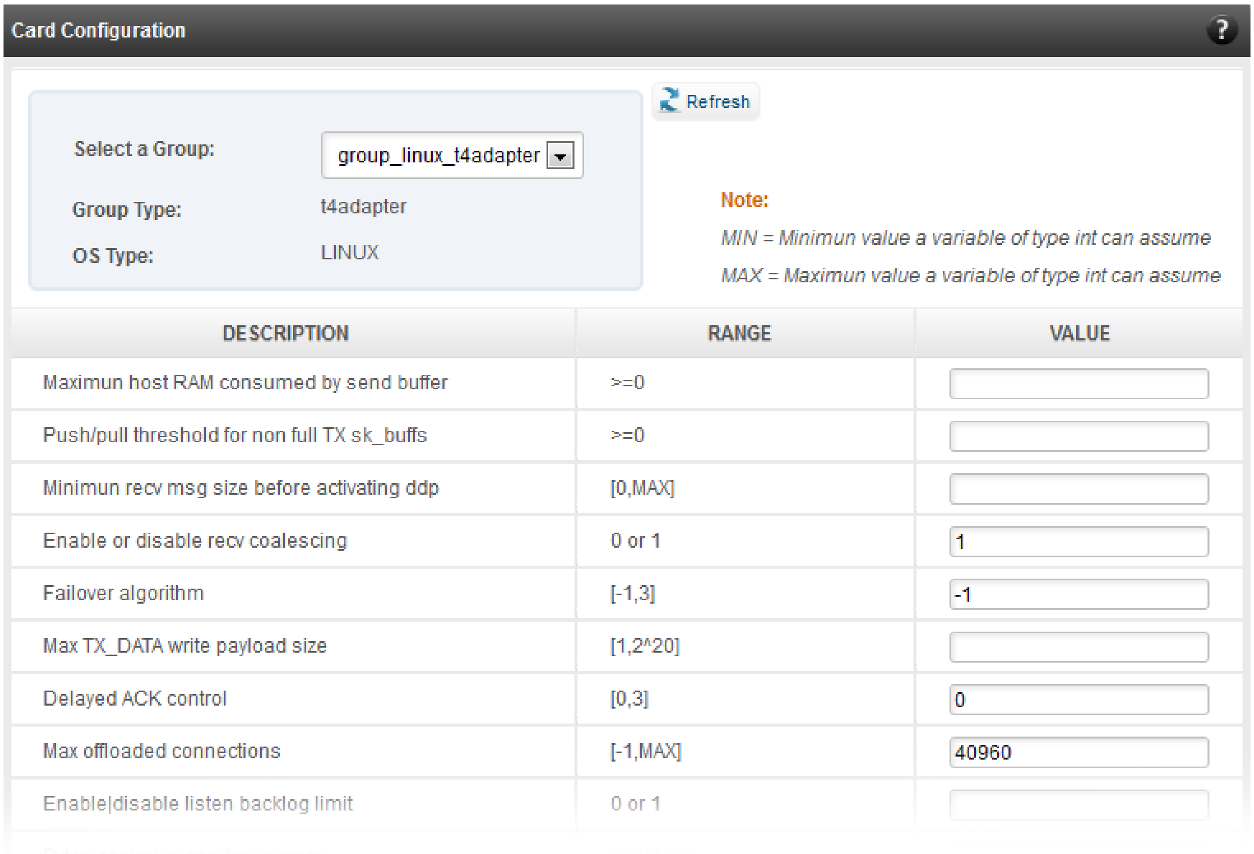


Figure 7.4.6 (e) – Card Configuration module

Port Configuration

In the **Port Configuration** module, you can set various port settings like enabling Tx checksum and TCP segmentation offload, setting Link speed and link duplex mode, etc. The settings depend on the device driver installed.

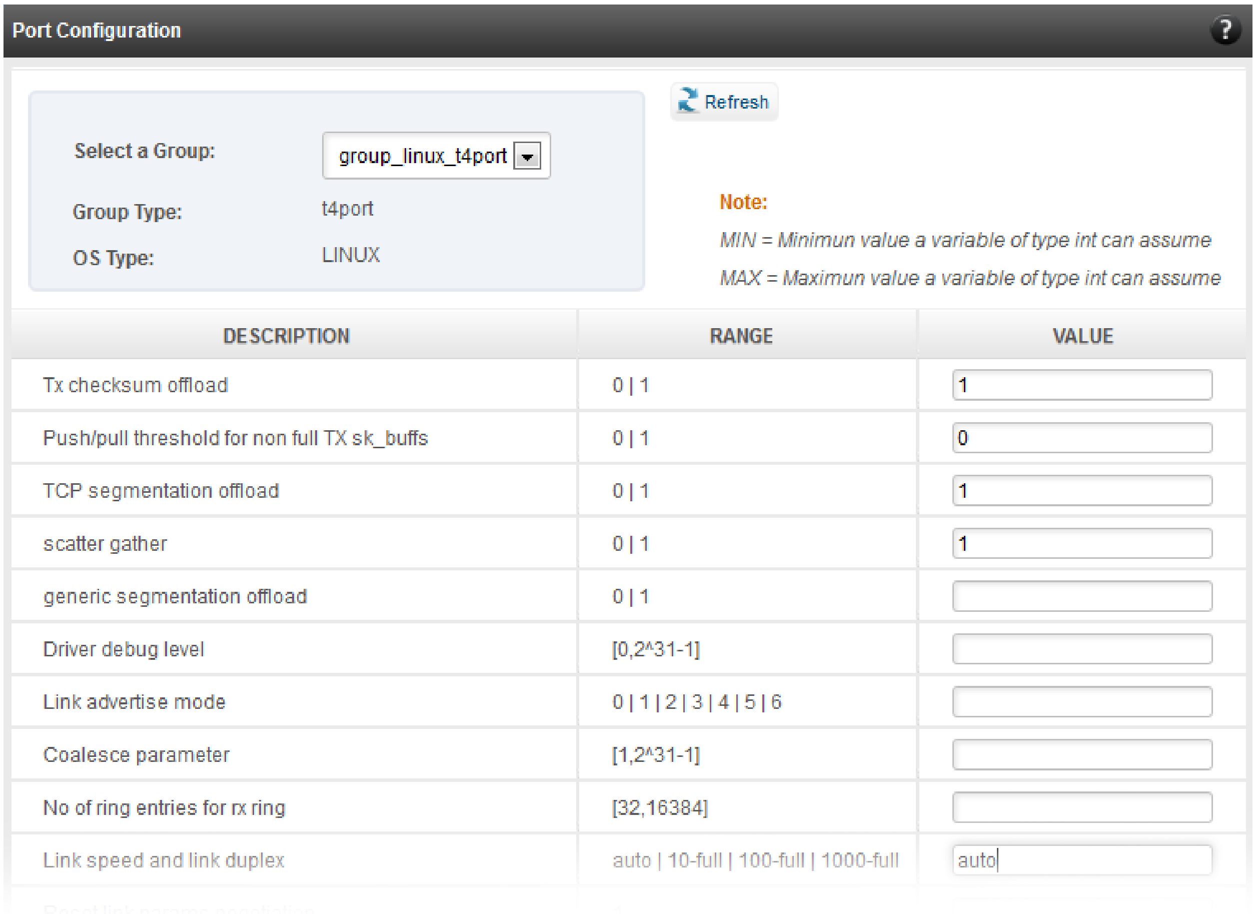


Figure 7.4.6 (f) - Port Configuration module

Bypass Configuration

Use the **Bypass Configuration** module to configure Chelsio's bypass adapters like B420-SR and B404-BT.

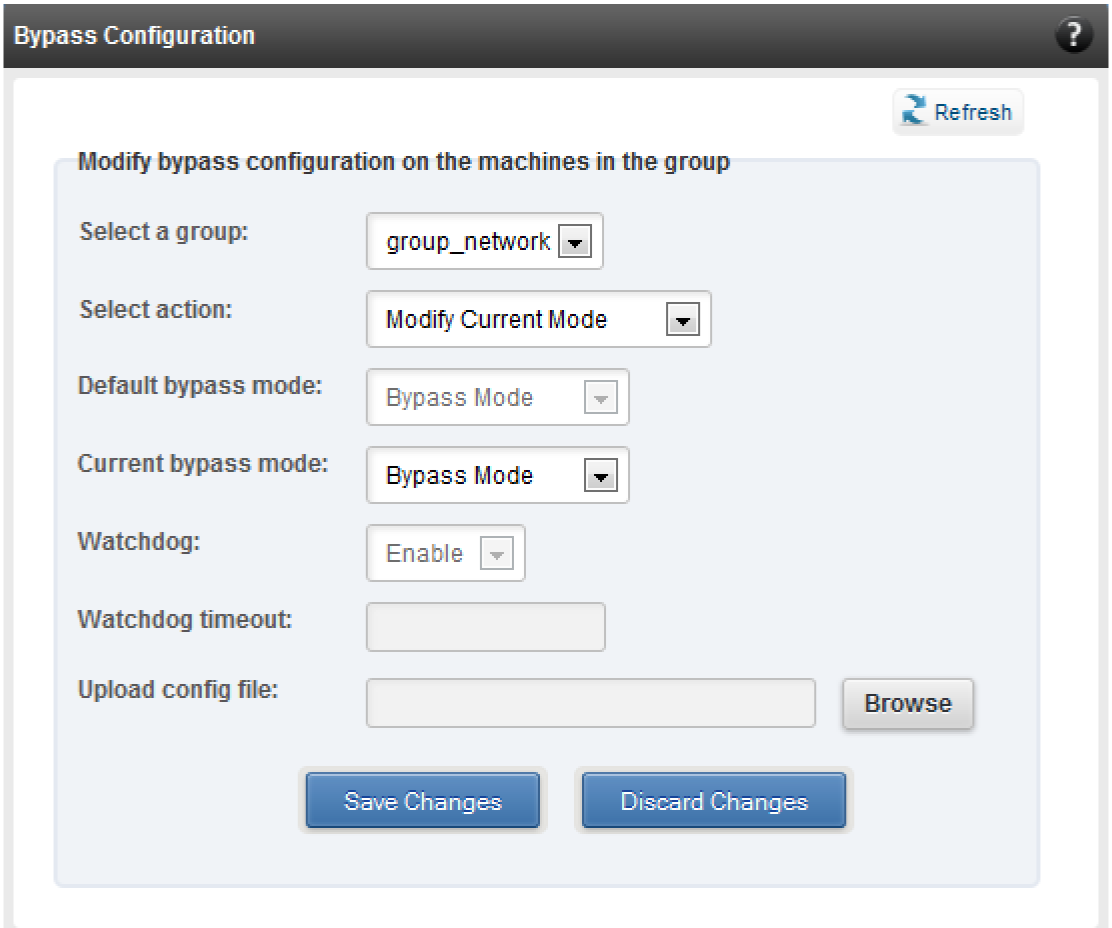


Figure 7.4.6 (g) - Bypass Configuration module

7.5. System page

The system page is displayed, when the system hostname / IP address is selected in the tree menu on the left. On adding a system, this item is automatically selected, and this page is displayed. The system page contains generic system and support modules which are discussed below:

7.5.1. System Summary

This module lists the system Hostname, Operating System, platform and also gives the count of the Chelsio cards found.

System Summary

PROPERTY	VALUE
Hostname	shark1.asicdesigners.com
Connected IP:Port	10.193.184.62:35001
Chelsio Cards	1
Operating system	Red Hat Enterprise Linux Server release 6.3 (Santiago)
Platform	x86_64 (amd64 / x64)

Figure 7.5.1 - System Summary module

7.5.2. Drivers Installation

Using this module, one can install various Chelsio drivers for different operating systems. You can choose the configuration file type (Linux Agents only).

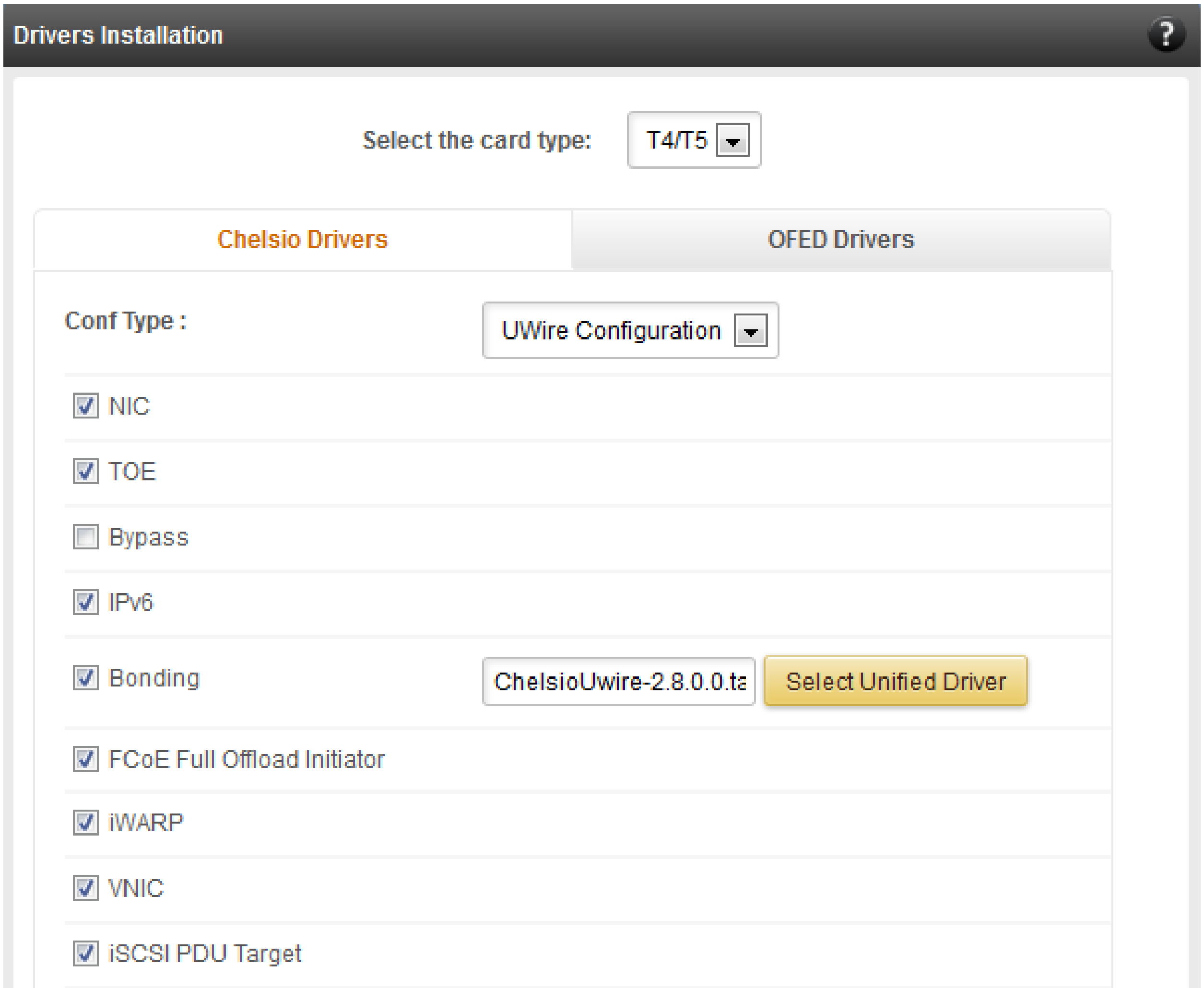


Figure 7.5.2 (a) - Drivers Installation module connected to Linux Agent

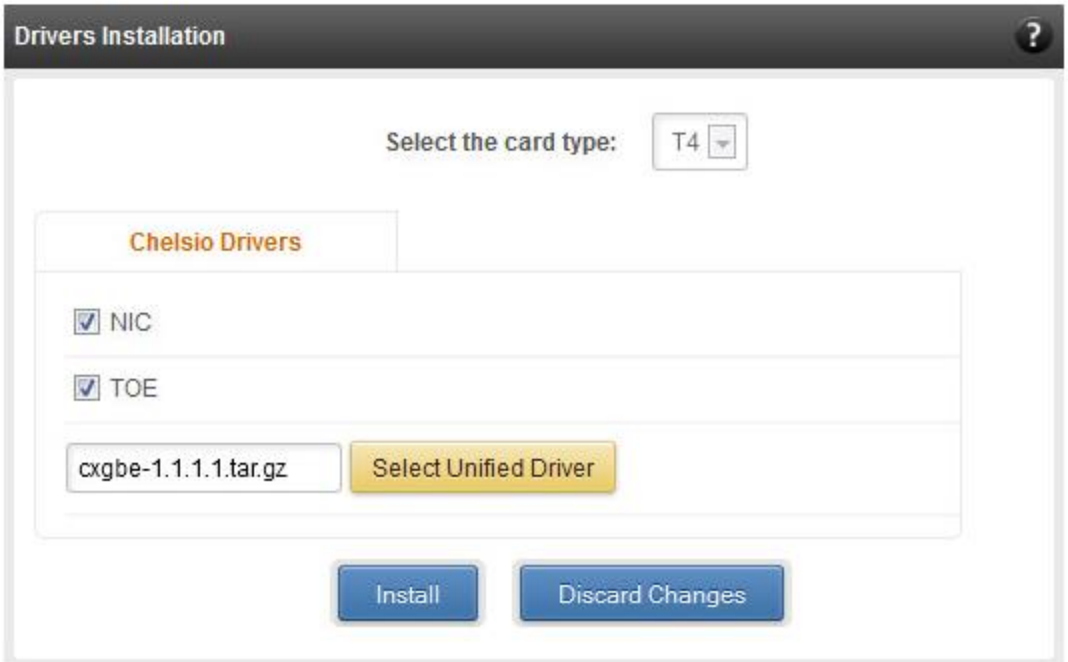
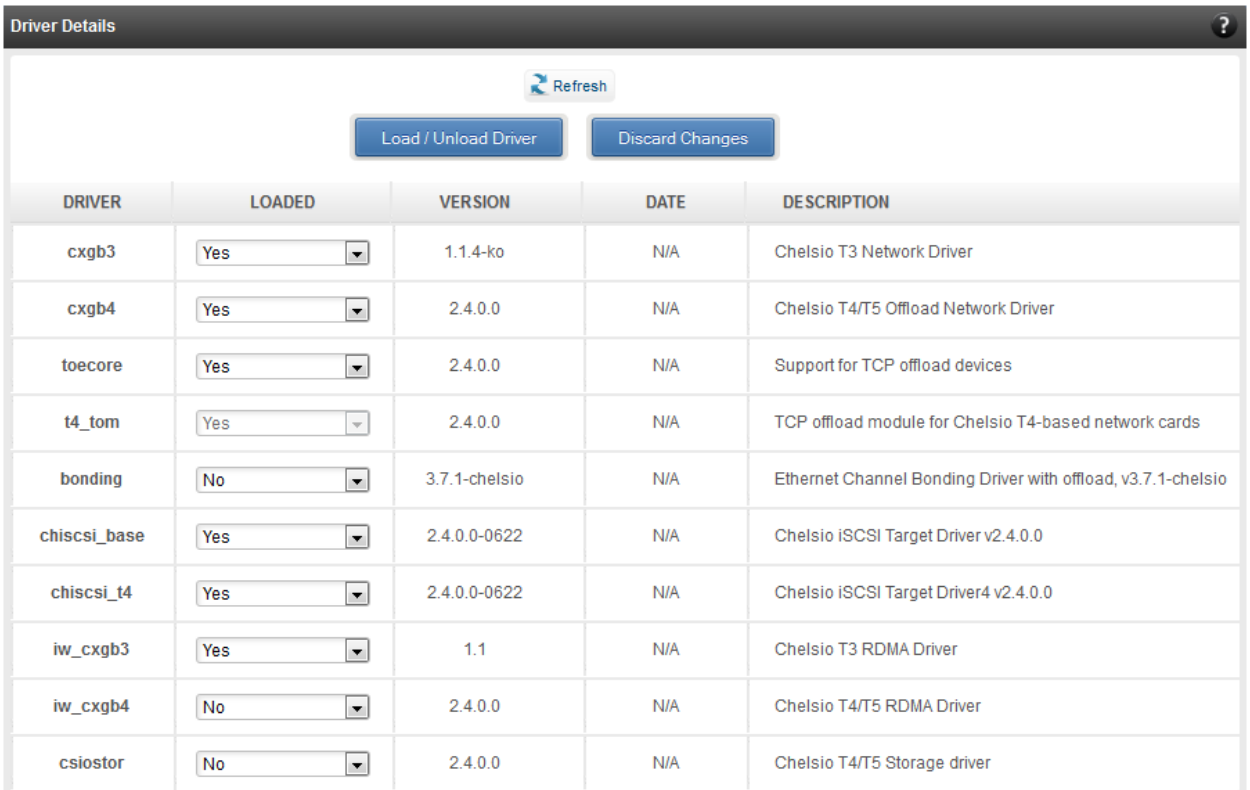


Figure 7.5.2 (b) - Drivers Installation module connected to FreeBSD Agent

7.5.3. Driver Details

A list of Chelsio device drivers with related information like driver description, version, current load status and installation date is shown in this module. To load or unload a particular driver, select the appropriate option (*Yes* to load, *No* to unload) in the corresponding cell of the *Loaded* column. To reload a driver select *Reload*. Finally click **Load/Unload Driver** button.

Click **Refresh** if changes are not reflected immediately. To reject the load/unload option selected, click **Discard Changes**.


Driver Details

Refresh

Load / Unload Driver Discard Changes

DRIVER	LOADED	VERSION	DATE	DESCRIPTION
cxgb3	Yes	1.1.4-ko	N/A	Chelsio T3 Network Driver
cxgb4	Yes	2.4.0.0	N/A	Chelsio T4/T5 Offload Network Driver
toecore	Yes	2.4.0.0	N/A	Support for TCP offload devices
t4_tom	Yes	2.4.0.0	N/A	TCP offload module for Chelsio T4-based network cards
bonding	No	3.7.1-chelsio	N/A	Ethernet Channel Bonding Driver with offload, v3.7.1-chelsio
chiscsi_base	Yes	2.4.0.0-0622	N/A	Chelsio iSCSI Target Driver v2.4.0.0
chiscsi_t4	Yes	2.4.0.0-0622	N/A	Chelsio iSCSI Target Driver4 v2.4.0.0
iw_cxgb3	Yes	1.1	N/A	Chelsio T3 RDMA Driver
iw_cxgb4	No	2.4.0.0	N/A	Chelsio T4/T5 RDMA Driver
csiostor	No	2.4.0.0	N/A	Chelsio T4/T5 Storage driver

Figure 7.5.3 - Driver Details module

7.5.4. System Diagnostics

Using this module, you can run various diagnostic tests on Chelsio adapters to troubleshoot adapter related issues. Select the adapter(s) from the list for which you want to run the test, select the operation (type of test; you can run more than one test at a time) and click **Run Test**. After the tests are completed, the results will be displayed in a tabular format.

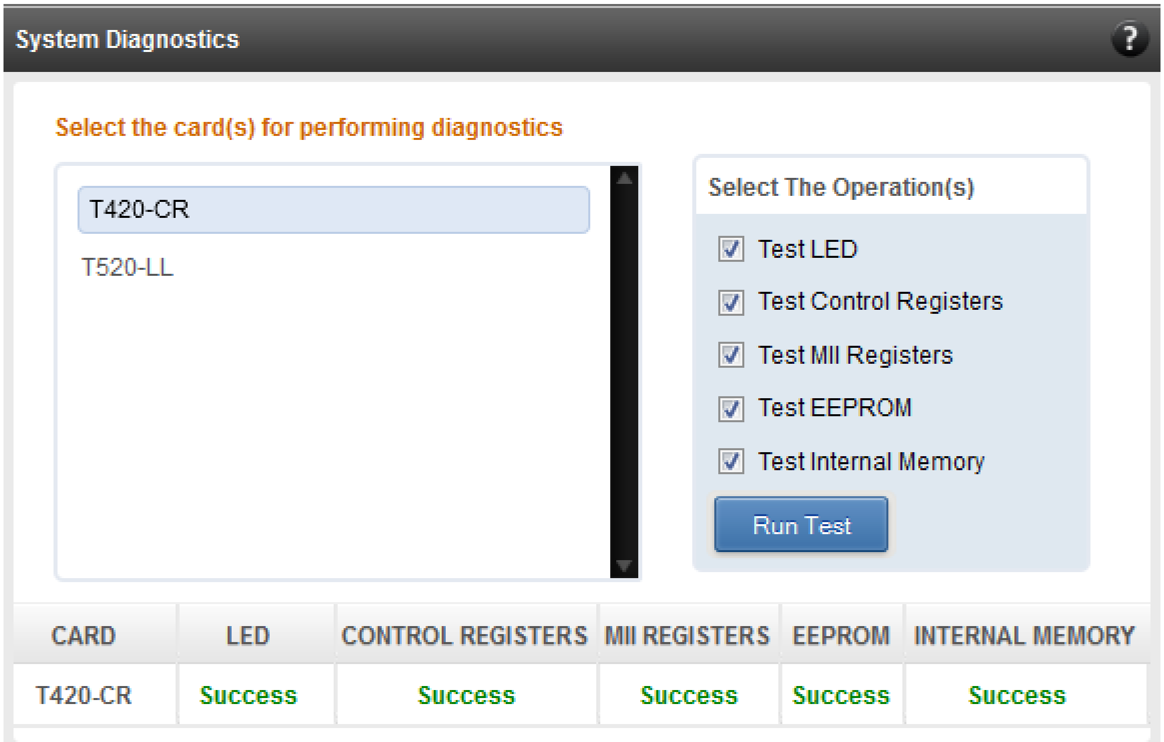


Figure 7.5.4 - System Diagnostics module for a T4 CNA

7.5.5. Unified Wire Manager Component Versions

A list of the Unified Wire Manager agent components installed on the managed system is shown in this module. The versions of the components are useful in case of reporting an issue to support.

Unified Wire Manager Component Versions

COMPONENT	VERSIONS
Server	2.4.33
libchlinopinitr	2.0.2
libchlintarget	2.0.4
chlintmfltr	2.0.1
libchlinnet	2.0.13
libchlinteam	2.0.3
libchlinhw	2.0.6
libchlinosapi	2.2.1

Figure 7.5.5 - Unified Wire Manager Component Versions module

7.5.6. KVM Configuration (Linux)

This module allows you to enable or disable KVM related operations. Once enabled, two related modules, **VM Configurations** and **VF Configurations**, will be available and you can manage VMs and related settings. While enabling, you will have to provide the number of Virtual Functions (VFs) you want to create for the physical ports.

The KVM Configuation module will be available only when Chelsio Network driver (*cxgb4*) and KVM modules (*kvm* and *kvm_intel/kvm_amd*) are loaded. If not done already, access Agent CLI and follow these steps to do so:

i. If any or all the aforementioned drivers are already loaded, unload them before proceeding:

```
[root@host~]# rmmod <kvm_intel/kvm_amd>
[root@host~]# rmmod kvm
[root@host~]# rmmod cxgb4
```

ii. Next, reload them using *modprobe*:

```
[root@host~]# modprobe kvm allow_unsafe_assigned_interrupts=1
[root@host~]# modprobe <kvm_intel/kvm_amd>
[root@host~]# modprobe cxgb4
```

Loading the *kvm* module with *allow_unsafe_assigned_interrupts=1* option enables use of device assignment without interrupt remapping support. This is required in order to asssign VFs to VMs.

iii. Finally, access WebGUI. Remove the Agent using the “Remove System” button and connect to it again from the **Bookmarks and History** module. You should now be able to see the KVM Configuration module.

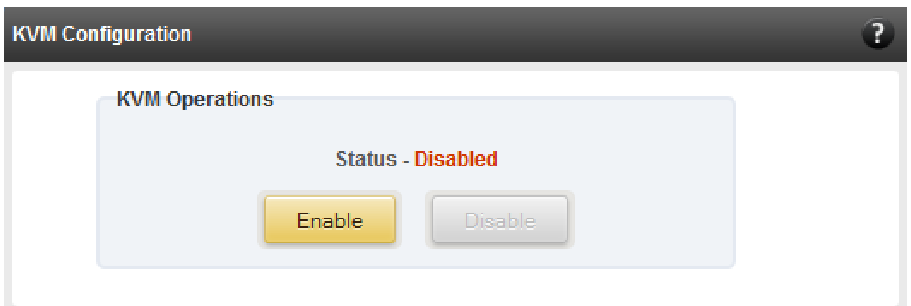


Figure 7.5.6 (a) – KVM Configuration module

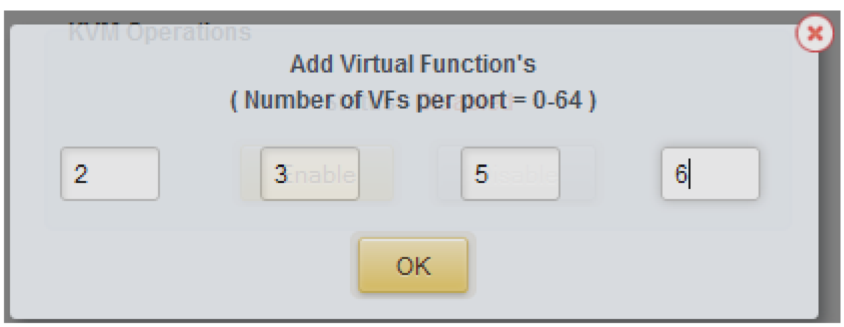


Figure 7.5.6 (b) – KVM Configuration module: Adding Virtual Functions

7.5.7. VM Configurations (Linux)

The VM Configurations module allows you to view UUID and domain state of Virtual Machines. You can perform various system power options like start or resume (if VM is paused), turn off, restart or suspend (pause) a VM.

You can perform similar actions on multiple virtual machines. To do so, click on the machine names in the list. The properties box will display the domain state of the machines selected. Now, click on any of the system power actions provided at the bottom.

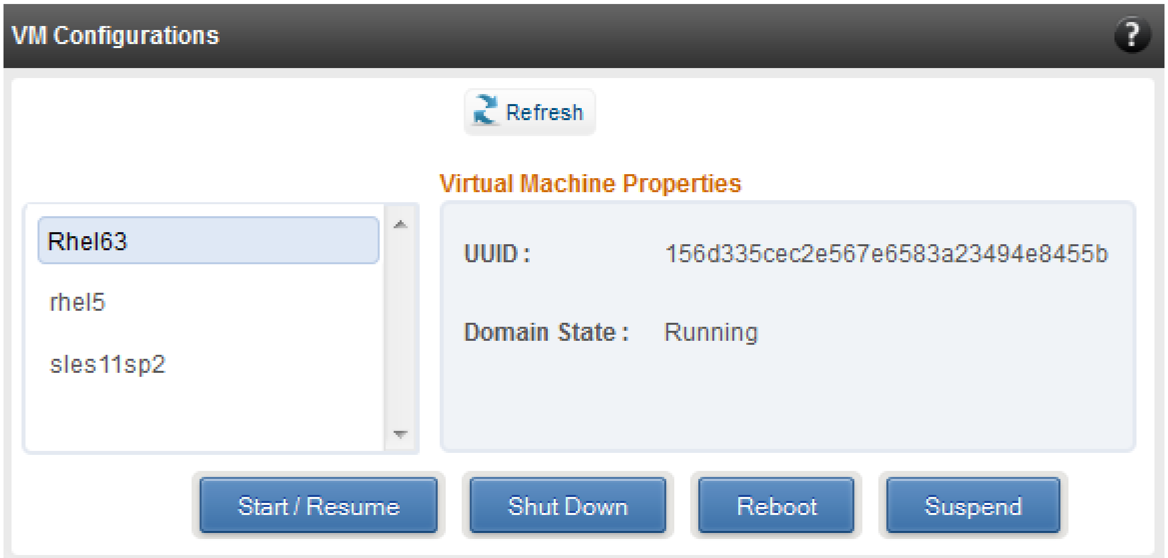


Figure 7.5.7 (a) – VM Configurations module

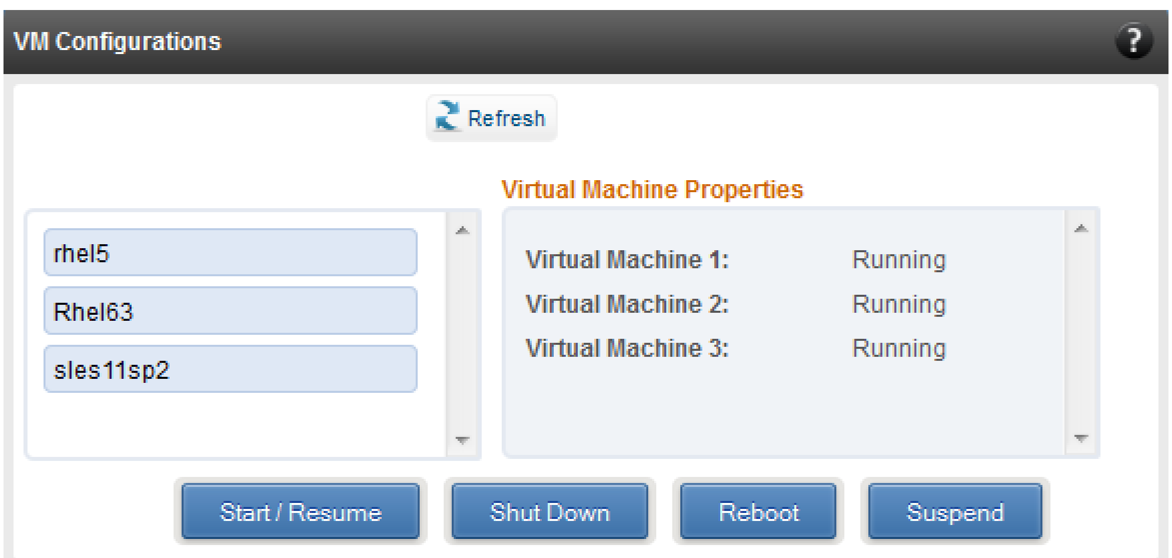


Figure 7.5.7 (b) – VM Configurations module: viewing properties for multiple VMs

7.5.8. VF Configurations (Linux)

The VF Configurations module lists all the VMs, Virtual Functions mapped to each Virtual Machine and all the available VFs. You can also add and remove VFs for a particular VM.

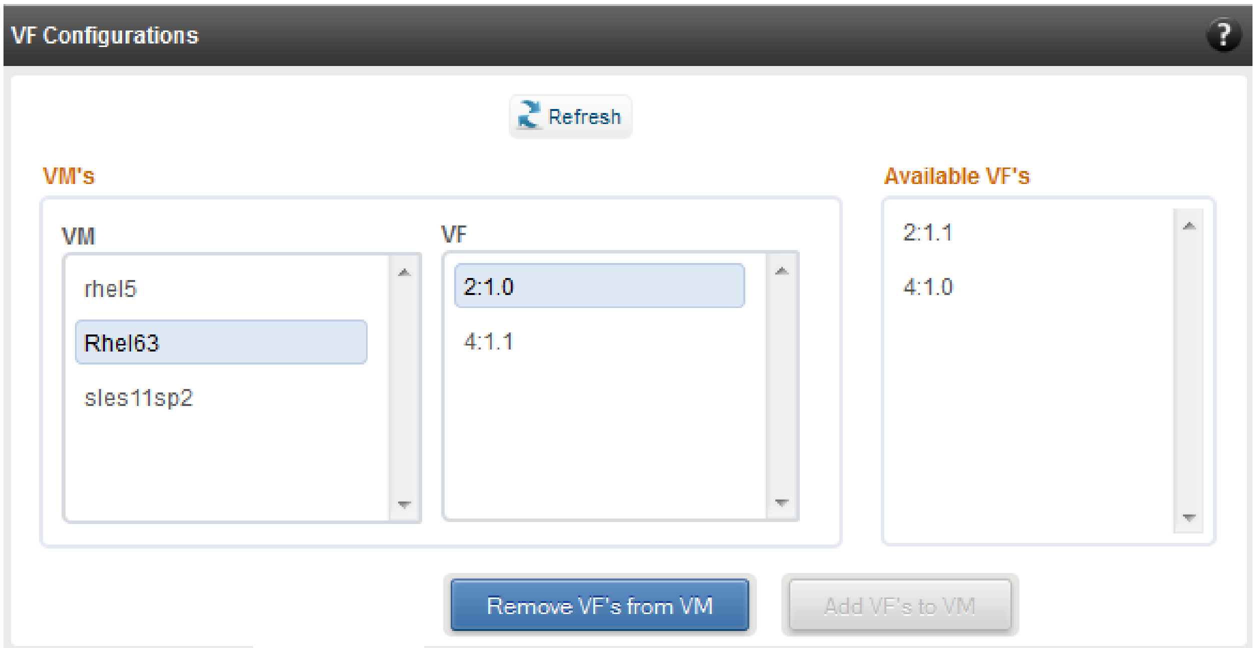


Figure 7.5.8 - VF Configurations module

7.5.9. Xen Configurations

The Xen Configurations module allows you to view UUID, power state of Virtual Machines and Virtual Functions assigned to them. You can perform various system power options like start, resume (if VM is paused), turn off, restart or suspend (pause) a VM.

You can perform similar actions on multiple virtual machines. To do so, click on the machine names in the list. The properties box will display the power state of the machines selected. Now, click on any of the system power actions provided at the bottom.

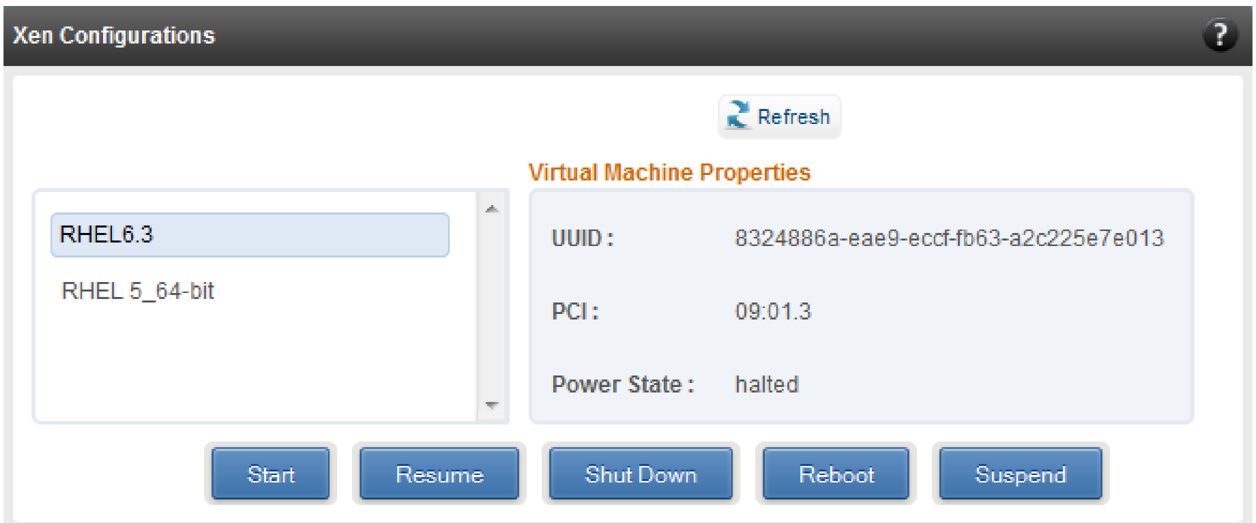


Figure 7.5.9 (a) – Xen Configurations module

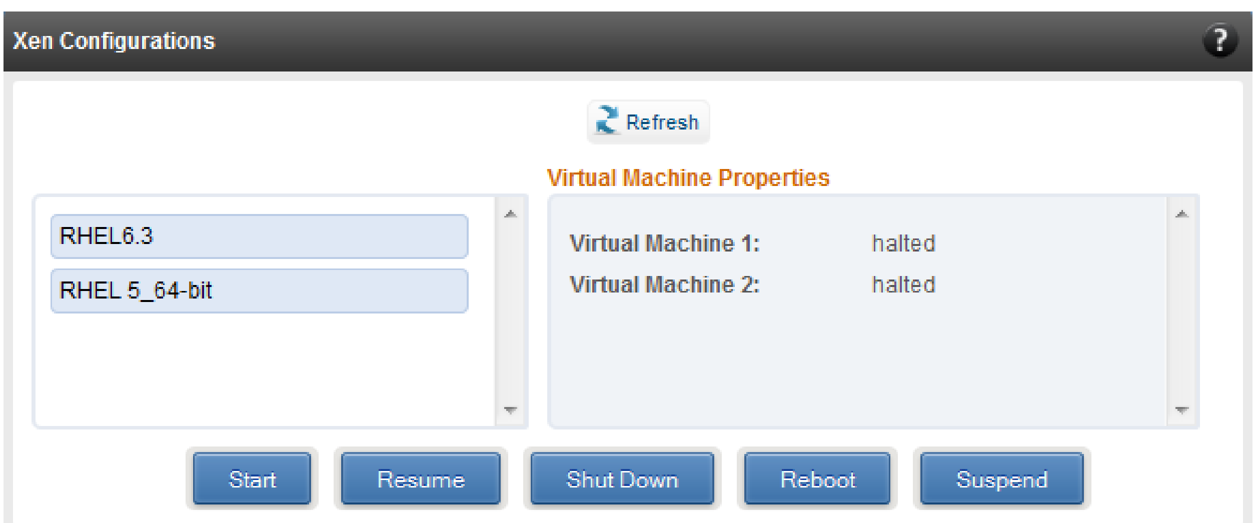


Figure 7.5.9 (b) – Xen Configurations module: viewing properties for multiple VMs

7.5.10. Xen VF Properties

Here you can view the list of virtual machines and list of available VFs. To assign a VF to a VM, select the guest name on the left and select the VF to be assigned on the right. You can assign more than one VF at a time. Finally, click "Assign VF" to add the selected VFs to the hosts.

To enable SR-IOV support, IOMMU must be enabled. To do this, select "Enable" for IOMMU and then click "Set IOMMU". Reboot the host machine for changes to take effect.

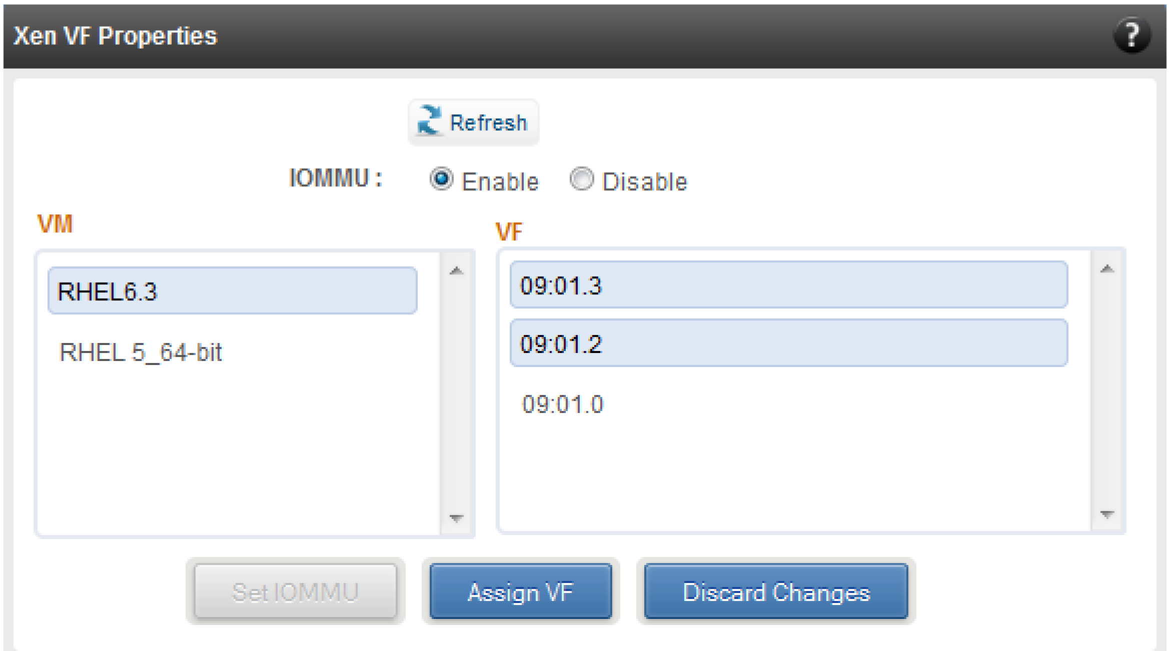


Figure 7.5.10 (a) – Xen VF Properties module

Important *Please ensure that Chelsio Network driver (cxgb4) with Virtual Functions is enabled in the* ***Driver Details*** *module before attempting to assign VF. Next, you will have to provide the number of Virtual Functions (VFs) you want to create for the physical ports. If VFs are not listed, unload the driver, reload it and create VFs again.*

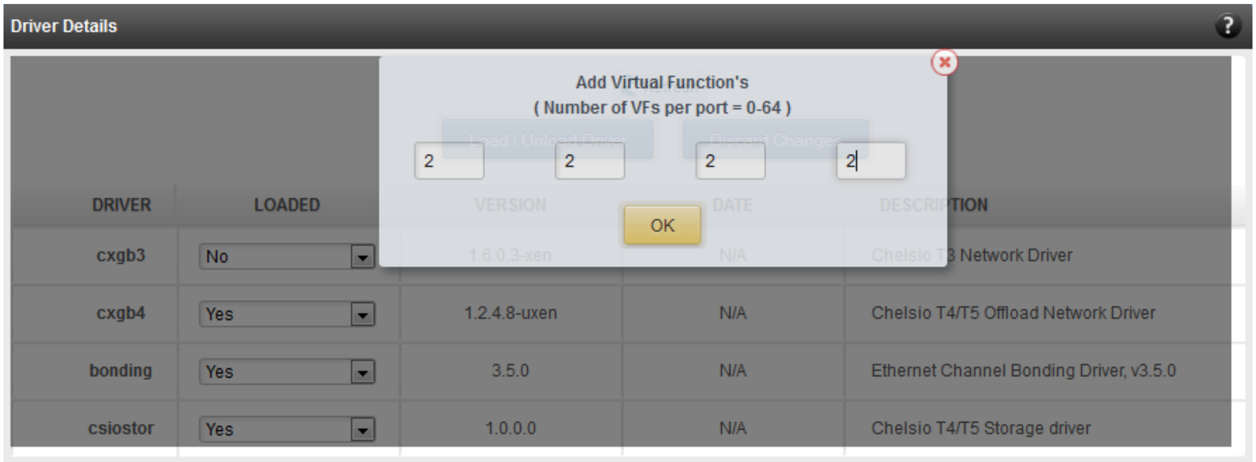


Figure 7.5.10 (b) – Xen VF Properties module: Adding Virtual Functions

7.5.11. Managed system application logs

The management agent logs its activities and any errors that occur, in */var/log/chelsio* in *Linux* and FreeBSD and in the Event log, in Windows. This log can be obtained in this module. Only 20 entries can be obtained and viewed at a time. Logs can be viewed by either choosing from a list of fixed range or by specifying a custom starting point.

Use the **Get Logs** button to retrieve, and **Hide Logs** button to clear the log entries. The **Delete Logs** button will remove the logs permanently from the agent.

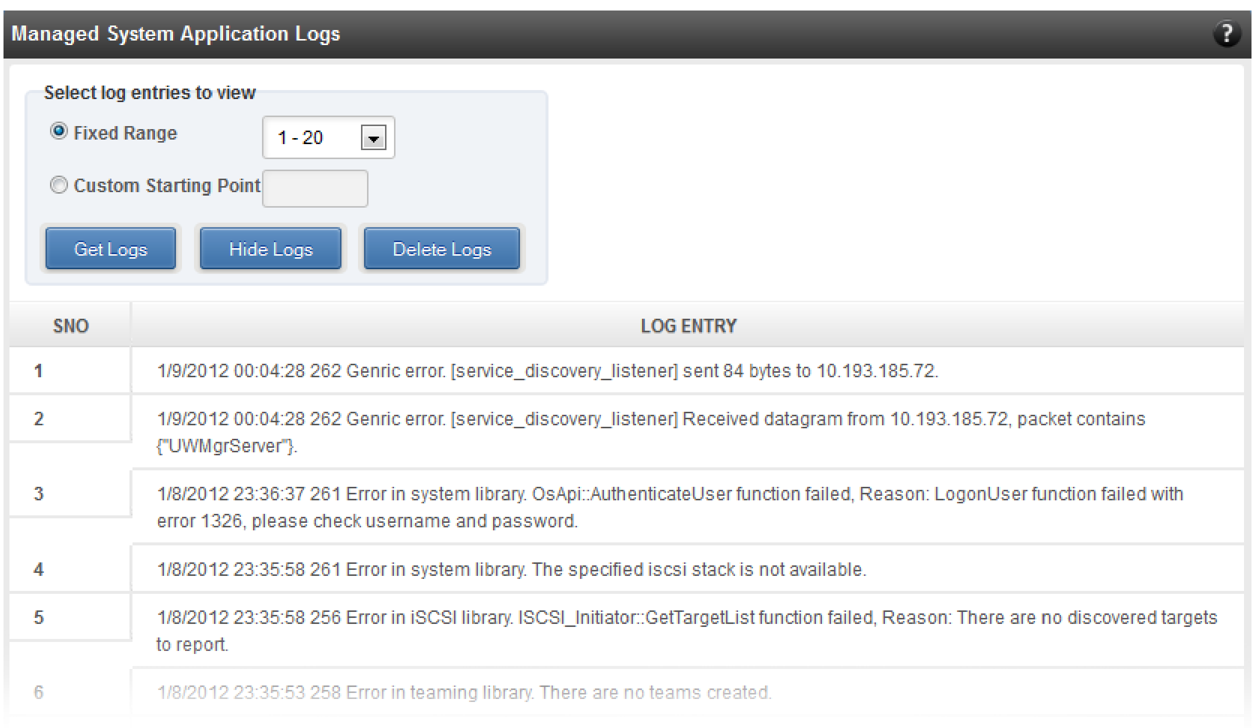


Figure 7.5.11 (a) - Managed System Application Logs module for Windows Agent

7.6. Network page

7.6.1. Network summary

The **Network Summary** module provides the total number of Chelsio adapters present, including the number of T5, T4 and T3 adapters. It also provides the total number of Network interfaces including corporate and Chelsio interfaces and VLANs.

Network

PROPERTY	VALUE
No. of Chelsio Cards	1
No. of T5/T4 Cards	1
No. of T3 Cards	0
No. of Network Interfaces	7

Figure 7.6.1 (a) – Network Summary module

7.6.2. Chelsio card page

When a Chelsio card is selected in the tree menu on the left, this page is displayed. It provides details of the card and associated settings. It also displays any card specific statistics that the hardware provides. The modules available on this page are as below:

- **Card summary**

This module provides PCI, firmware and other details of the card. The card's serial number and factory MAC address are also provided for inventory purposes.

Card Summary

PROPERTY	VALUE
PCI Vendor ID : Device ID	1425 : 440e
PCI Bus Location (Bus : Device : Function)	04:00.4
Card Serial Number	PT34110046
Factory MAC Address	00:07:43:11:51:C0
Firmware Version	1.9.23.0, TP 0.1.9.1
Ethernet Ports	4
Offload Support	OffloadCard
Connector	10G SFP+

Figure 7.6.2 (a) - Card Summary module

TCP Offload settings (Linux & FreeBSD)

The TCP offload settings applicable to the card are shown here. These settings are only available when using the TOE capable drivers (*t3_tom* and *toecore* for T3 adapters; *t4_tom* and *toecore* for T4 and T5 adapters). On changing the settings, the changed settings may not reflect immediately on refreshing the data. Highlight the system item in the tree menu on the left, and click **Refresh**, to refresh data from the system, in case the updated settings are not being shown.

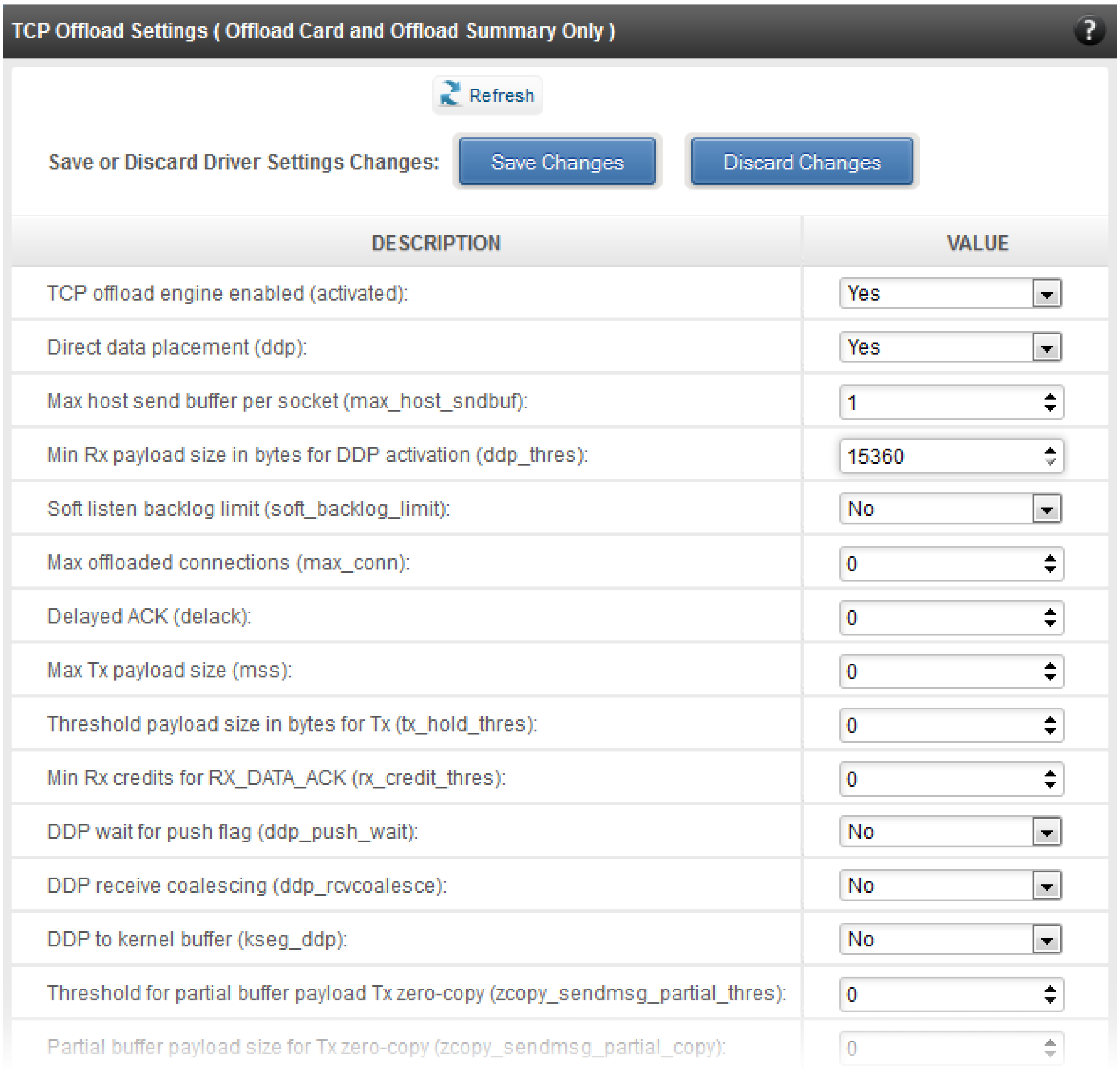


Figure 7.6.2 (b) – TCP Offload Settings module for a Linux Agent

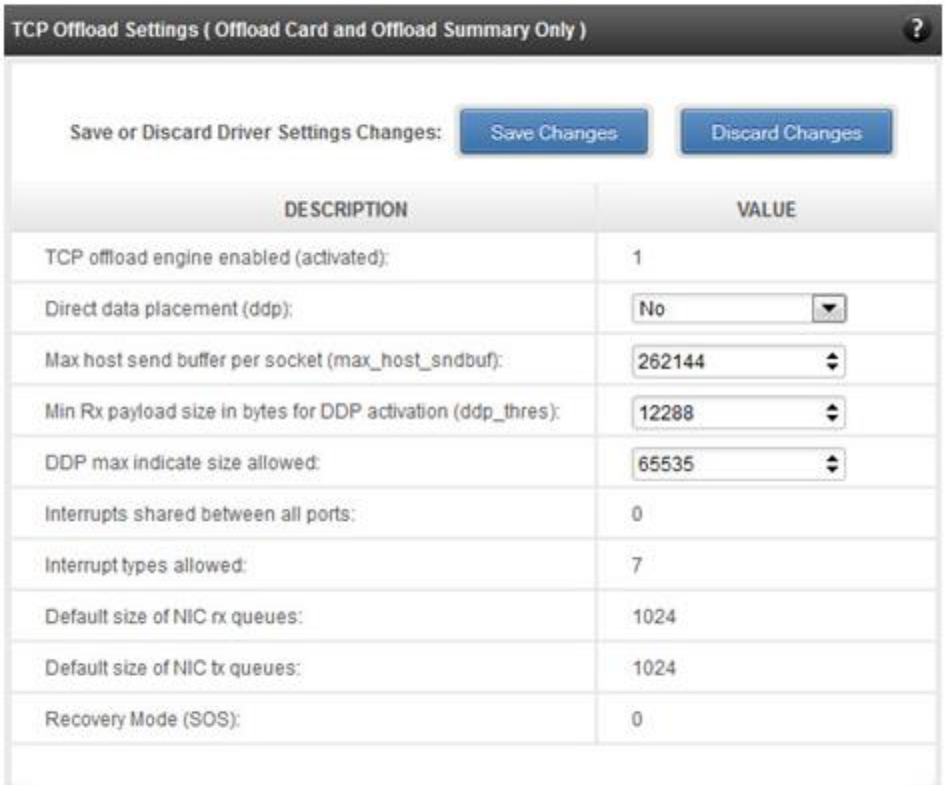


Figure 7.6.2 (c) - TCP Offload Settings module for a FreeBSD Agent

Device Driver settings (Windows)

The device driver settings applicable to the card are shown here. For Chelsio T5 and T4 adapters, only the *MaxVMQueues* field will be displayed. On changing the settings, the changed settings may not reflect immediately on refreshing the data. Highlight the system item in the tree menu on the left, and click **Refresh**, to refresh data from the system, in case the updated settings are not being shown.

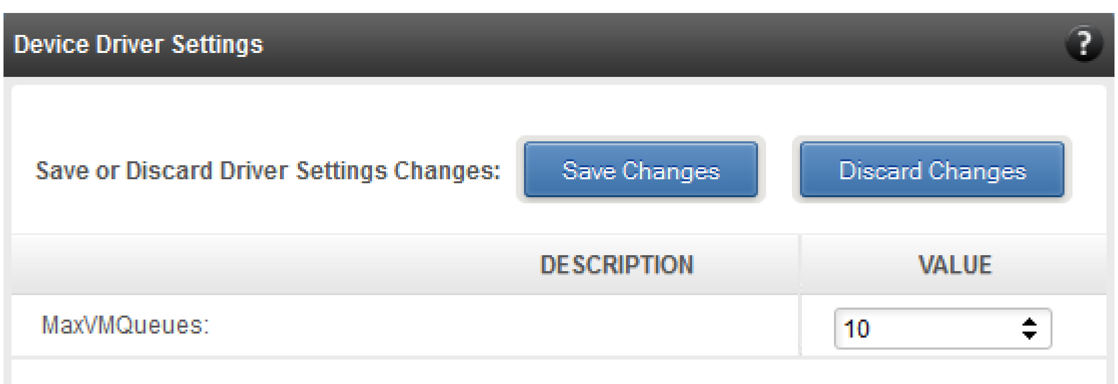


Figure 7.6.2 (d) – Device Driver Settings module for a Windows Agent

Card statistics

Certain statistics are maintained on a per card basis (instead of a per port basis), since the card has a TCP/IP offload capability. The statistics are for TCP and IP protocol processing done in the card's hardware. These statistics may only be applicable if the card is TOE enabled.

Card Statistics

STATISTIC	VALUE
ipInReceive	15
ipInHdrErrors	0
ipInAddrErrors	0
ipInUnknownProtos	0
ipInDiscards	0
ipInDelivers	15
ipOutRequests	265
ipOutDiscards	0
ipOutNoRoutes	0
ipReasmTimeout	0
ipReasmReqds	0
ipReasmOKs	0
ipReasmFails	0

Figure 7.6.2 (e) - Card Statistics module for a T3 HBA

7.6.2.1. Chelsio card's port

The port page is displayed on selecting a port of a Chelsio card listed in the tree menu on the left. It provides details of the port and port settings. It also displays any port specific statistics that the hardware provides. The modules available on this page are as below:

- **Port summary**

Port details such as the Ethernet adapter nam, link status, etc are shown in this module.

Port Summary

PROPERTY	VALUE
Port Name	eth44
Link	Link up
Port Supported	N/A
Port Type	Direct Attach Copper
Supported link modes	10000baseT/Full
Advertised link modes	Not reported
PauseautoNeg	No
Transceiver	External
Advertised Auto-negotiation	No
Ring_max_rx	16384
Ring_max_rx_mini	16384
Ring_max_rx_jumbo	0

Figure 7.6.2.1 (a) - Port Summary of T5/T4 CNA on Linux Agent

Port settings

Port settings such as MTU, Link speed and others can be set in this module. The settings depend on the device driver installed.

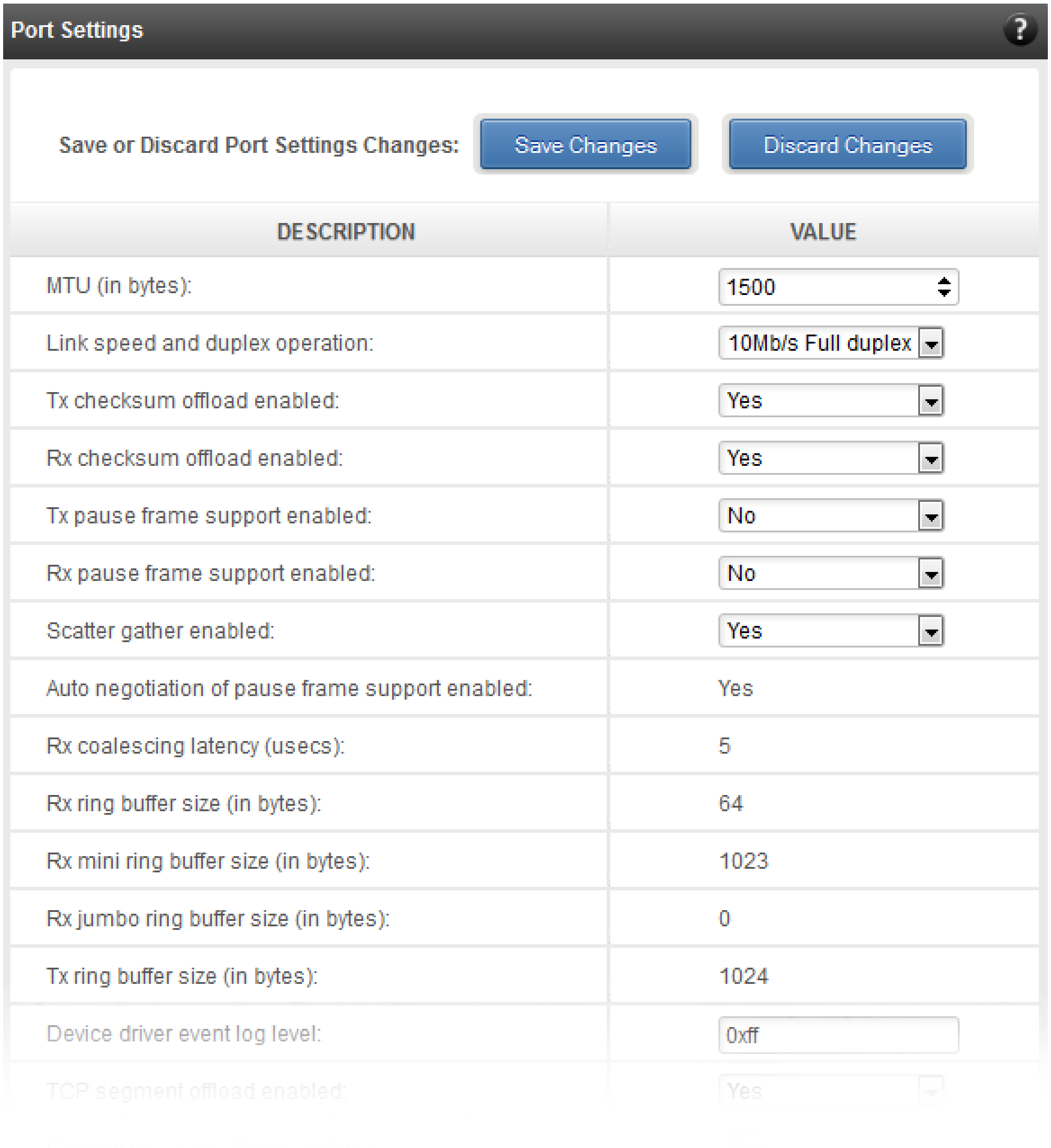


Figure 7.6.2.1 (b) - Port Settings of T4/T5 CNA on Linux Agent

Port statistics

Ethernet statistics and additional hardware statistics for the port are displayed in this module.

Port Statistics

STATISTIC	VALUE
TxOctetsOK	16208
TxFramesOK	137
TxBroadcastFrames	5
TxMulticastFrames	132
TxUnicastFrames	0
TxErrorFrames	0
TxFrames64	0
TxFrames65To127	132
TxFrames128To255	0
TxFrames256To511	5
TxFrames512To1023	0
TxFrames1024To1518	0

Figure 7.6.2.1 (c) - Port Statistics of T4/T5 CNA on Linux Agent

7.6.3. Networking Management page

The system networking and teaming / bonding configurations are shown on this page. IP addresses, MTU, VLAN Ids, DNS and default gateway settings can be viewed and modified here. Network adapters can also be enabled or disabled as required. The modules available on this page are as below:

System Network configuration

The list of network adapters on the system is displayed in a list on the left. The icon for the adapter indicates whether it is administratively enabled and if it is connected to the network. Network teams are also indicated with an appropriate icon. The primary IP address (IPv4) can be set for the adapter, when it is selected. There is an option to add/modify/delete additional IP addresses or aliases for the specified adapter. Use the option to add additional IP addresses with caution, since multiple IP addresses configured on the same adapter, for the same network, may result in unpredictable behavior of the system's networking stack. Maximum Transfer Unit (MTU) can be set between 1500-9000 bytes. VLAN id can also be set for an adapter within the range 0-4094 (enter 0 to disable it).

You can use the View/Set IP addresses option to add, modify or delete IP aliases.

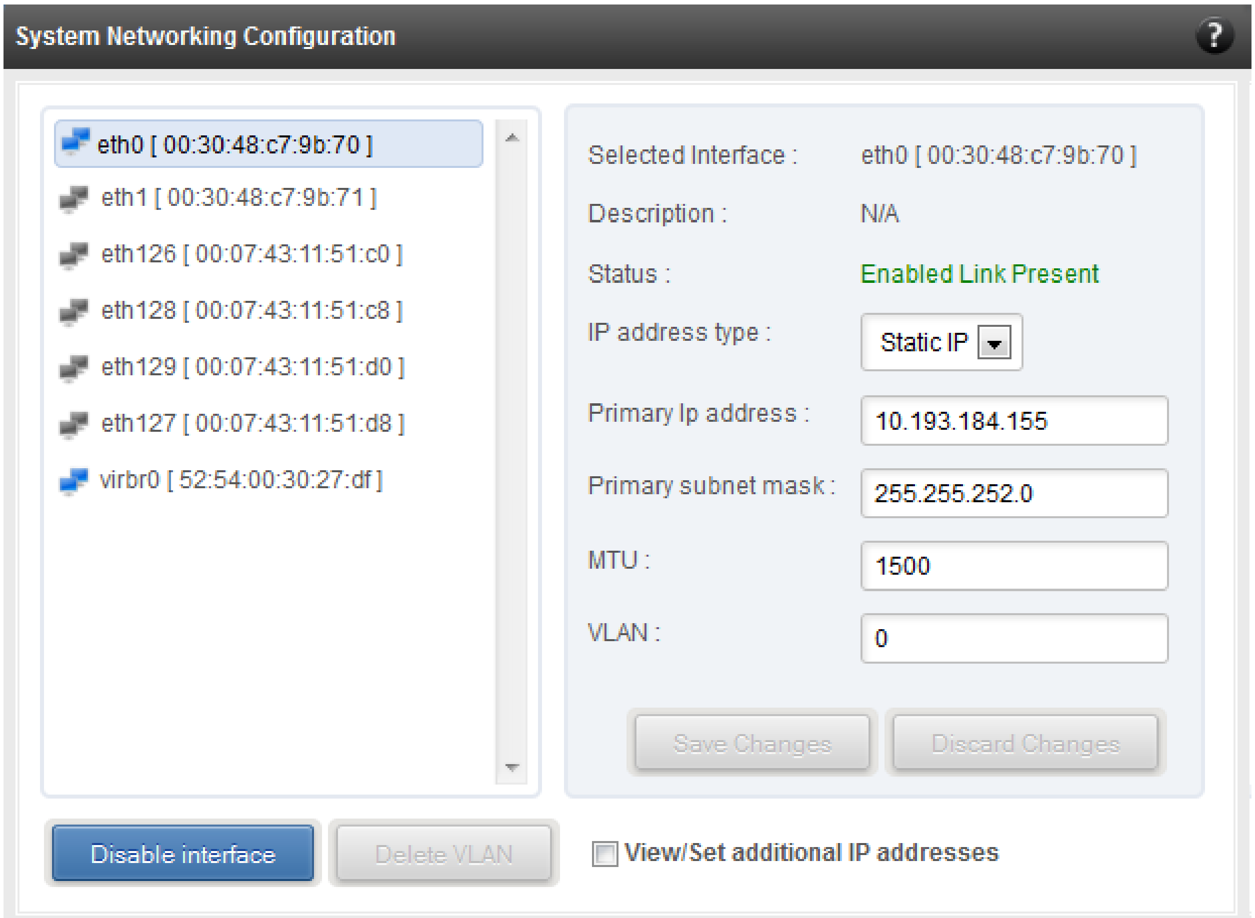


Figure 7.6.3 (a) - System network configuration module

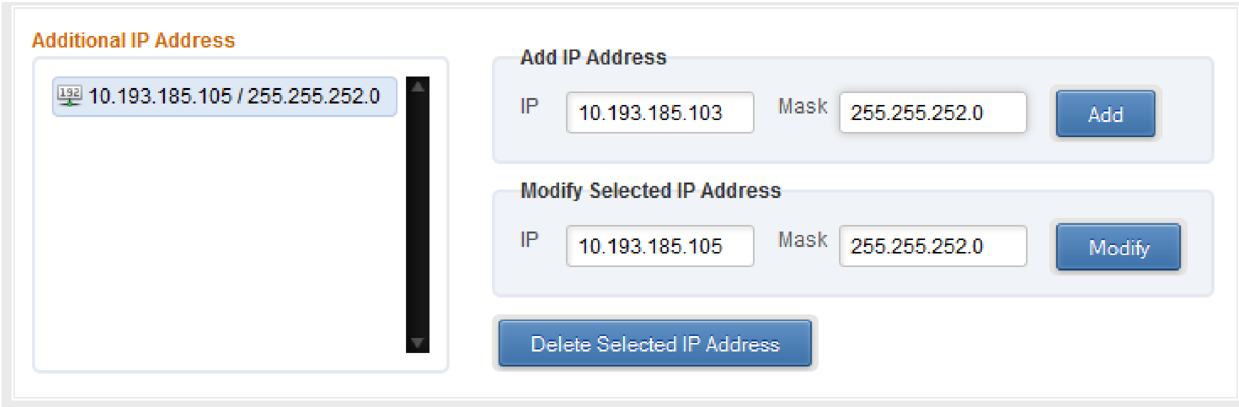


Figure 7.6.3 (b) – Managing IP aliases

System network statistics

Using this module, one can generate reports based on Throughput pkts/sec and Throughput Mbs (Receive, Transmit, Bi-direction) in Table and Graph format for a network adapter. A report for hardware statistics can be generated based on different parameters, only in the Table view in the **Advanced NIC characteristics**. The **polling time** field sets the average time (in seconds) based on which the table/graph updates the report.

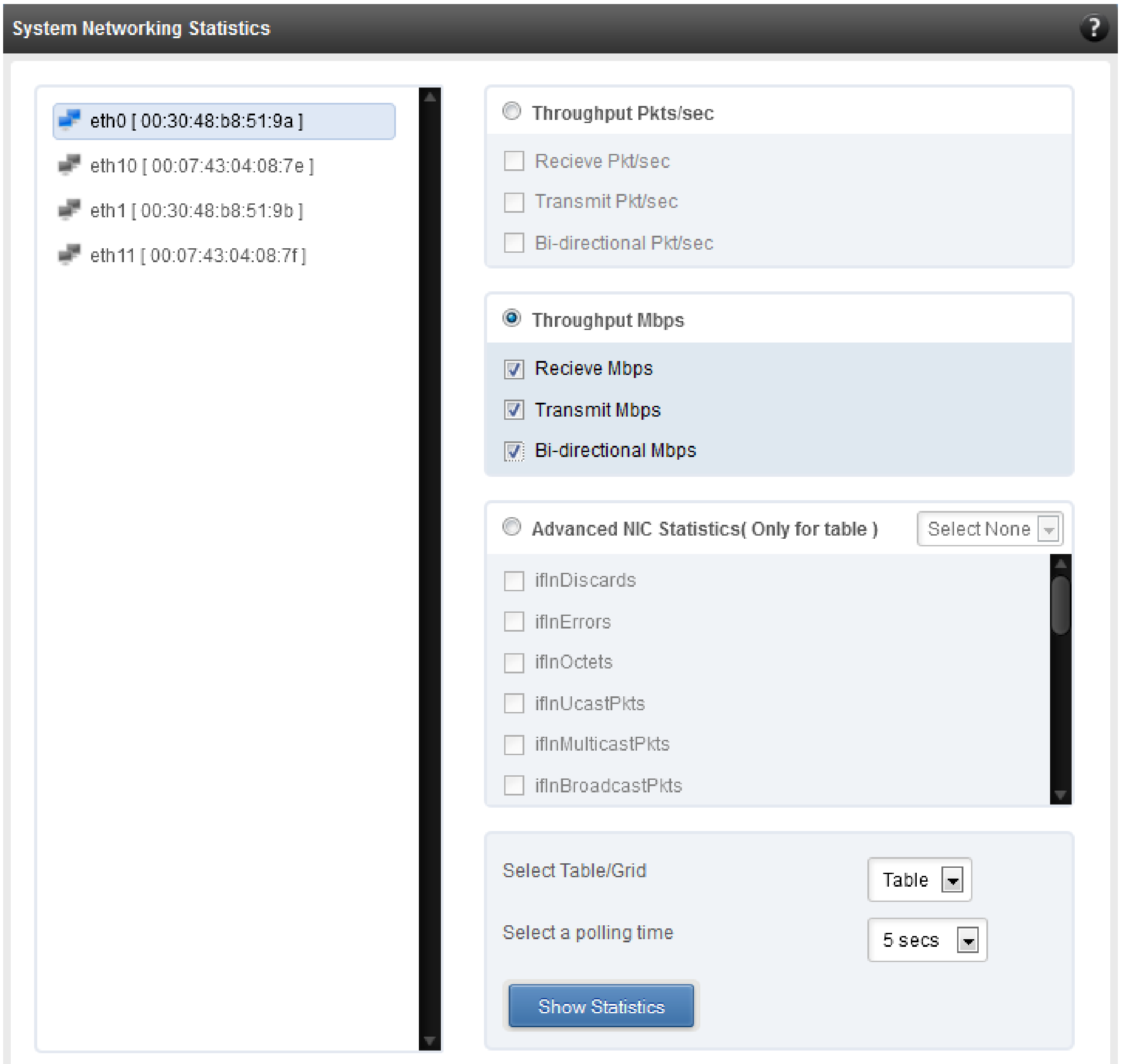


Figure 7.6.3 (c) - System network statistics module

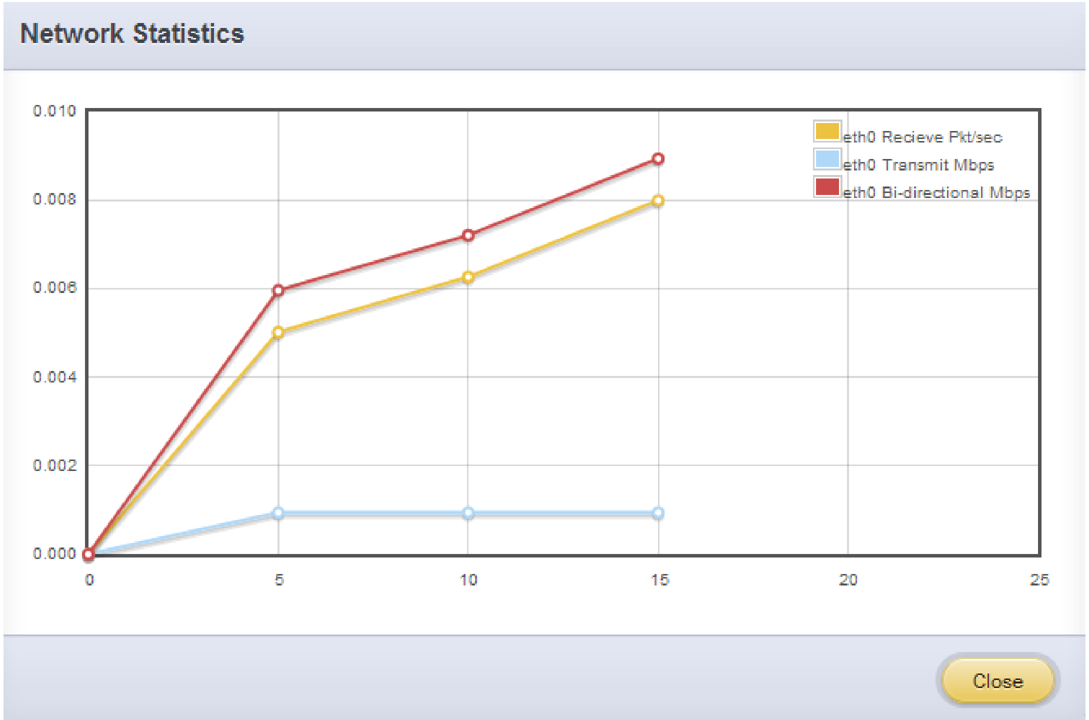
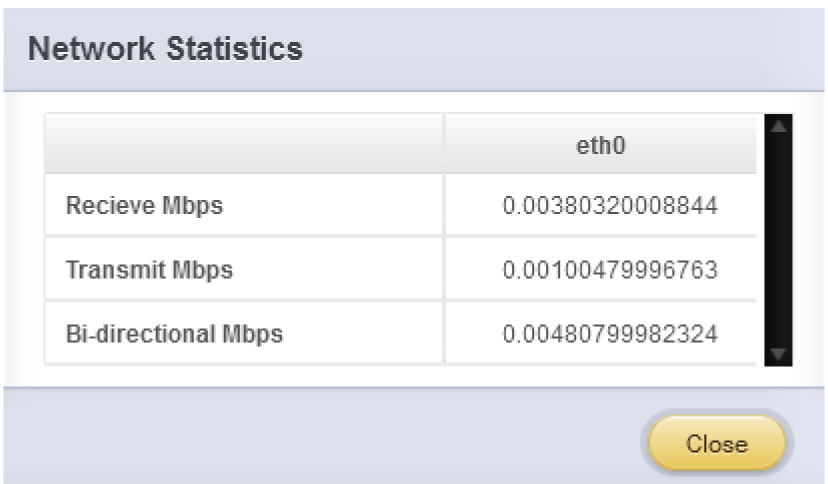


Figure 7.6.3 (d) - Network Throughput Vs Time instant Graph



	eth0
Recieve Mbps	0.00380320008844
Transmit Mbps	0.00100479996763
Bi-directional Mbps	0.00480799982324

Figure 7.6.3 (e) - Network Throughput Vs Time instant Table

Default Gateway and DNS configuration

The DNS servers list can be set here. The default gateway for remote networks and the Internet can also be set here. On Linux and FreeBSD, only one default gateway is allowed. On Windows, you may set multiple default gateways. Use the option to set multiple default gateways with caution, since it may cause the system to stop communicating with external networks.

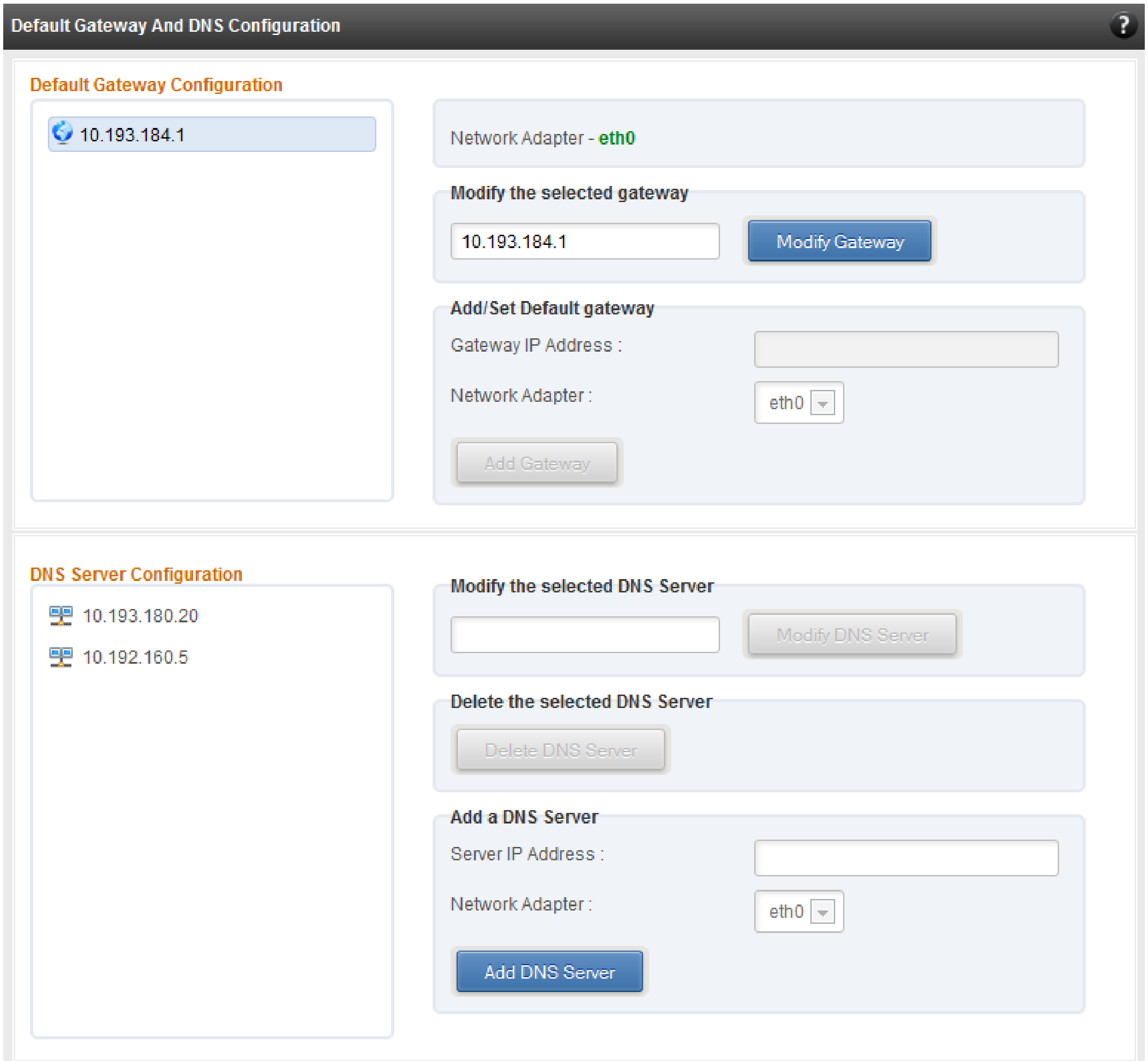


Figure 7.6.3 (f) - Default gateway and DNS configuration module for a FreeBSD Agent

Create a network team/bond device (Linux and FreeBSD)

A list of regular network adapters is provided here, to create a Network Team / Bond device. The available modes for the team depend on the OS teaming / bonding driver in use. On Linux the team may be created with a DHCP or Static IP address. On Windows, only DHCP is allowed when creating the team, although both DHCP and Static IP addressing is supported for the team adapter, after it is created successfully. Please check with the driver documentation for the supported modes for creating a team / bond, with offload enabled Chelsio cards. All modes may not be available with all configurations / combinations. Also, the team members can only be 2 ports of a single offload-enabled card, and not across Chelsio cards. Do not mix third party cards and offload-enabled Chelsio cards in a single team.

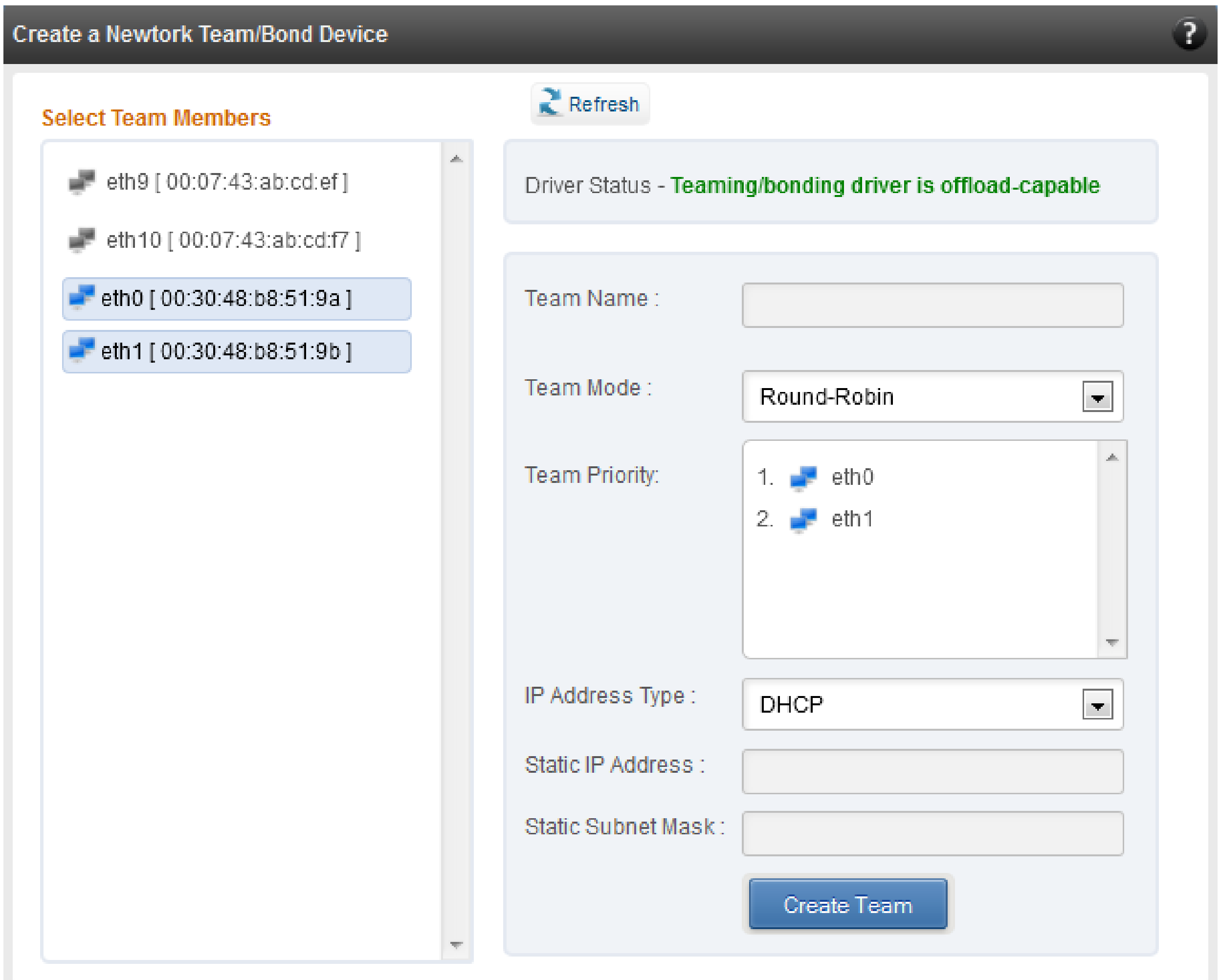


Figure 7.6.3 (g) - Create a network team/bond device module

Network troubleshooting

This module allows detecting and troubleshooting various network connectivity issues. The Ping utility helps to contact a system by specifying IP address, Number of ICMP packets to send and packet timeout. The result of the ping can be viewed by clicking on the **Ping Result** button.

Using **TraceRoute** one can determine the route taken by packets across an IP network.

Use the **GetConnections** utility to view currently active TCP/UDP connections. Offload status for each connection is also displayed if protocol offload hardware is available. This is useful for troubleshooting any connectivity issues for clients to various services.

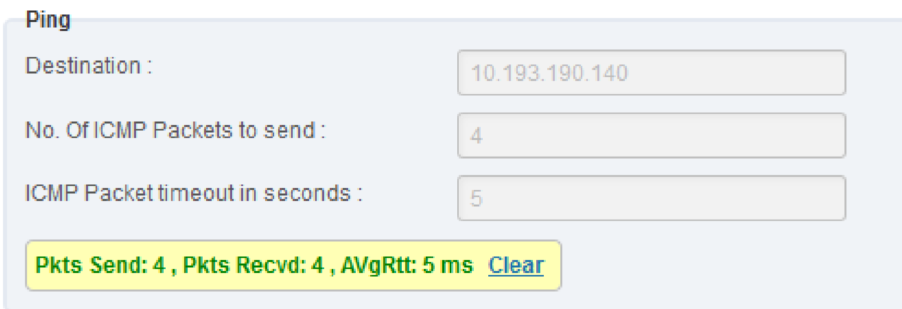


Figure 7.6.3 (h) - Ping Utility

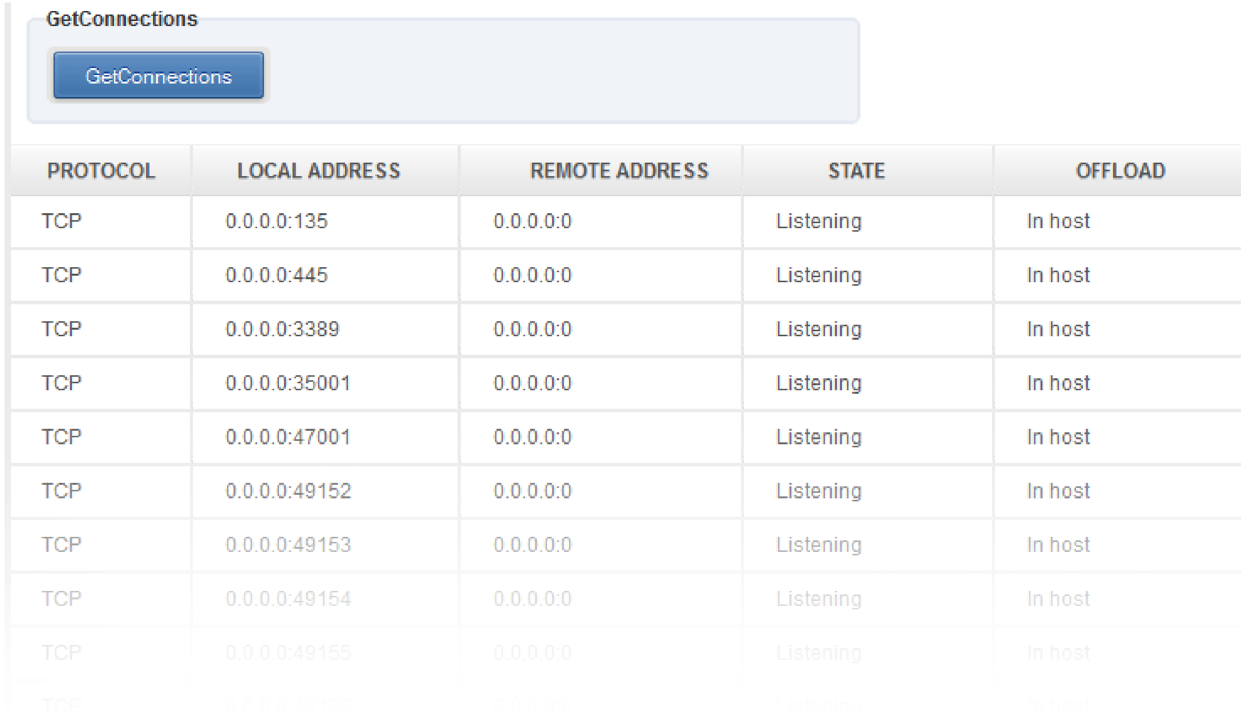
TraceRoute

Destination : www.chelsio.com

Hop Count	Round Trip Time	Ipv4 Address
1	2 ms	10.193.184.1
2	0 ms	10.193.177.3
3	5 ms	111.93.129.157
4	8 ms	121.241.196.101
5	3 ms	121.240.1.242
6	23 ms	172.29.250.33
7	24 ms	180.87.38.5
8	131 ms	80.231.217.17
9	130 ms	80.231.217.6
10	131 ms	80.231.154.17
11	132 ms	208.178.58.109
12	274 ms	208.178.63.114
13	272 ms	72.13.84.18

Hide

Figure 7.6.3 (i) - TraceRoute Utility



PROTOCOL	LOCAL ADDRESS	REMOTE ADDRESS	STATE	OFFLOAD
TCP	0.0.0.0:135	0.0.0.0:0	Listening	In host
TCP	0.0.0.0:445	0.0.0.0:0	Listening	In host
TCP	0.0.0.0:3389	0.0.0.0:0	Listening	In host
TCP	0.0.0.0:35001	0.0.0.0:0	Listening	In host
TCP	0.0.0.0:47001	0.0.0.0:0	Listening	In host
TCP	0.0.0.0:49152	0.0.0.0:0	Listening	In host
TCP	0.0.0.0:49153	0.0.0.0:0	Listening	In host
TCP	0.0.0.0:49154	0.0.0.0:0	Listening	In host
TCP	0.0.0.0:49155	0.0.0.0:0	Listening	In host

Figure 7.6.3 (j) - GetConnections Utility

7.6.3.1. Hypervisor

- #### Xen Bridge Configuration

The **Xen Bridge Configuration** module allows you to view and manage network bridges, virtual nterfaces (vifs) and virtual machines to which those virtual interfaces are assigned. The left pane displays a list of different bridges created. Clicking on a bridge name will display related properties on the right.

If a virtual interface is attached to a particular bridge, a “+” link appears next to the bridge name. Expanding the “+” link will display the virtual interface and expanding the “+” for that virtual interface displays the virtual machines to which it is assigned. Click on the virtual machine names to view their properties on the right.

To delete a bridge, click on the bridge name and then click “Delete Bridge”. This will also delete the virtual interface attached to that bridge and the VM(s) to which the interface was assigned.

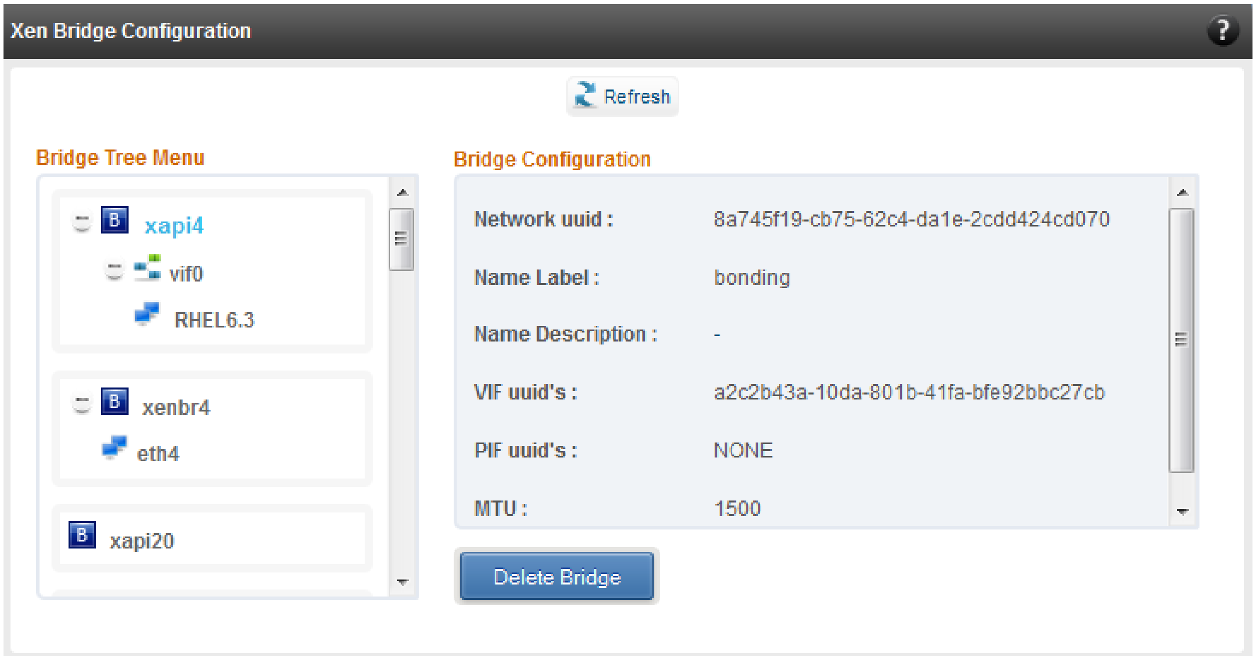


Figure 7.6.3.1 (a) – Xen Bridge Configuration module

- ## Bridge Configuration (Linux)

The Bridge Configuration module allows you to view and manage network bridges, virtual network interface (vnets) and virtual machines to which those virtual network interfaces are assigned. The left pane displays a list of different bridges created. Clicking on a bridge name will display related properties on the right.

If a virtual network interface is attached to a particular bridge, a “+” link appears for that bridge name. Expanding the “+” link will display the virtual network interface and expanding the “+” link for that virtual network interface displays the virtual machine to which it is assigned. Click on the virtual machine names to view their properties on the right. Only bridge properties are editable.

To detach a virtual network interface from a bridge, select it and click “Delete VIF/VNET“ Restart the guest machine for changes to take effect.

To delete a bridge, click on the bridge name and then click “Delete Bridge”. If there are virtual network interfaces attached, you will have to detach them first.

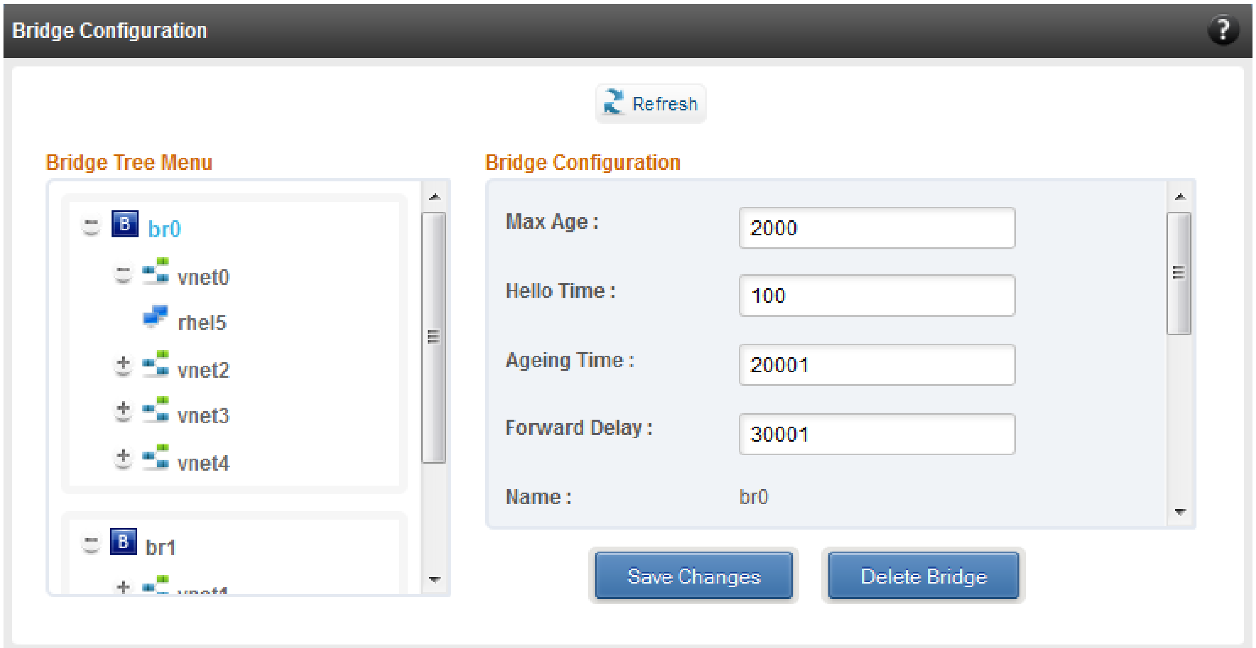


Figure 7.6.3.1 (b) – Bridge Configuration module (Linux)

Virtual Network Configuration (Linux)

Using the Virtual Network Configuration module, you can create network bridges and attach them to virtual machines. You can also assign physical interfaces on the host to bridges.

To create a bridge, enter a name and click “Create” in the “Create Bridge” section. All other parameters are optional. If not specified, the bridge will be created with default values. Except **STP** and **Priority**, you can change all of the other parameters in the **Bridge Configuration module** once the bridge is created.

Use the “Add Bridge to VM” section to attach a bridge to a virtual machine.

To assign interfaces on the host to a bridge, specify the bridge name and the host interface in the “Add Interface to Bridge” section. Click “Add”. You can add multiple interfaces to the same bridge using the aforementioned method. The **Cost** and **Priority** parameters are optional.

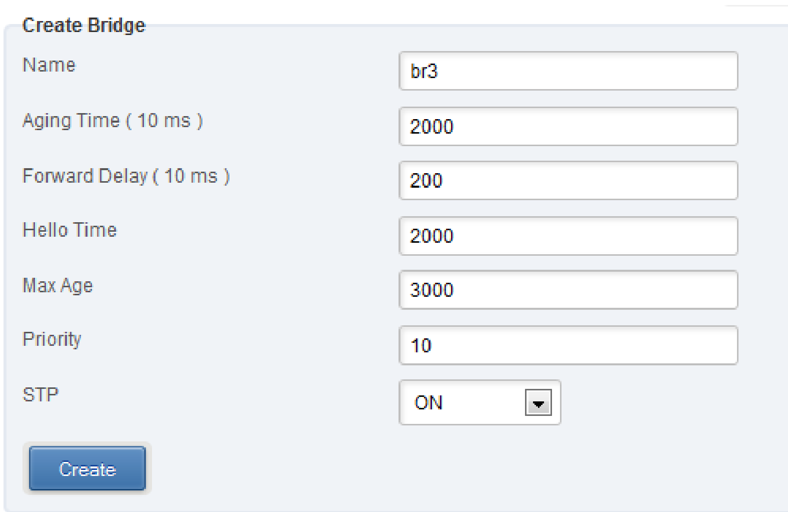


Figure 7.6.3.1 (c) – Creating Bridge

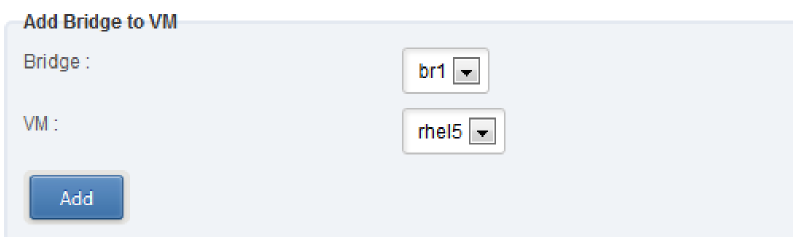


Figure 7.6.3.1 (d) – Adding Bridge to VM

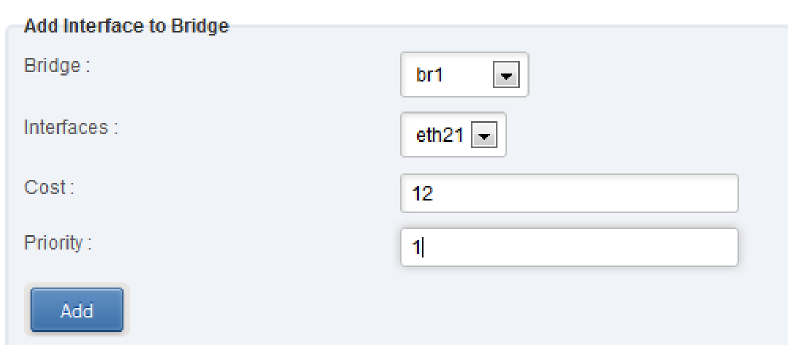


Figure 7.6.3.1 (e) – Adding interfafe to Bridge

Virtual Network Configuration (Xen)

Using the Virtual Network Configuration module, you can create network bridges. You can also create and attach virtual interfaces to them.

To create a bridge, enter a label for the bridge and click “Create”. The **MTU** and **Name Description** fields are optional. The bridge name will be generated automatically by the operating system. Once created, it will appear in the **Xen Bridge Configuration** module. You will have to use the label provided earlier to identify the bridge.

You can attach upto 64 virtual interfaces to a particular bridge. **Device Number**, **Bridge Name** and **VM Name** are the mandatory fields while creating a virtual interface.

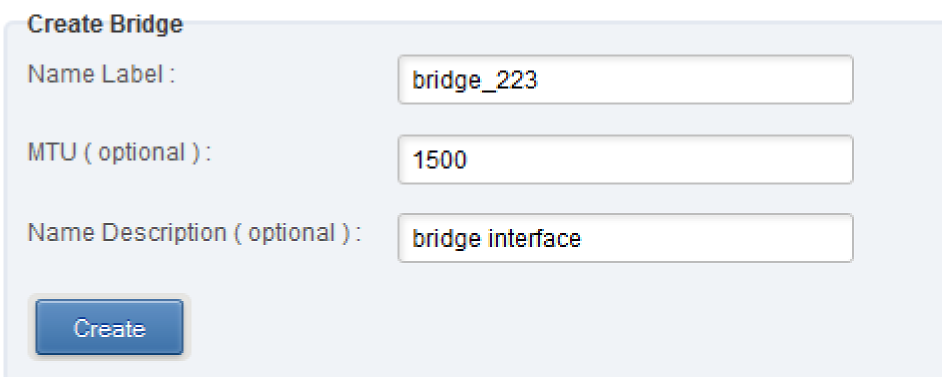


Figure 7.6.3.1 (f) – Creating Bridge

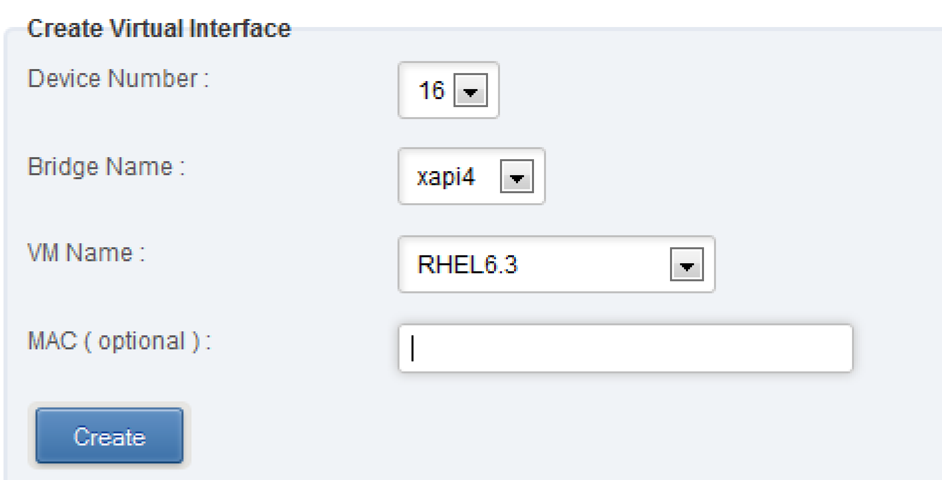


Figure 7.6.3.1 (g) – Creating and attaching virtual interface

Virtual Switch Configuration (Windows)

This module allows you to view and manage virtual networks. The left pane displays a list of different virtual networks created. Clicking on a virtual network name will display related properties on the right.

If a virtual network is added to a virtual machine, a "+" link appears next to the virtual network name. Expanding the "+" link will display the virtual machines to which the network is attached. Click on the virtual machine names to view their properties on the right.

To delete a virtual network, click on the network name and then click "Delete Switch". If it is attached to a virtual machine, you will have to detach the virtual machine first. To do so, click on the virtual machine and click "Detach". Similarly, detach all the virtual machines and then use the "Delete Switch" to delete the virtual network.

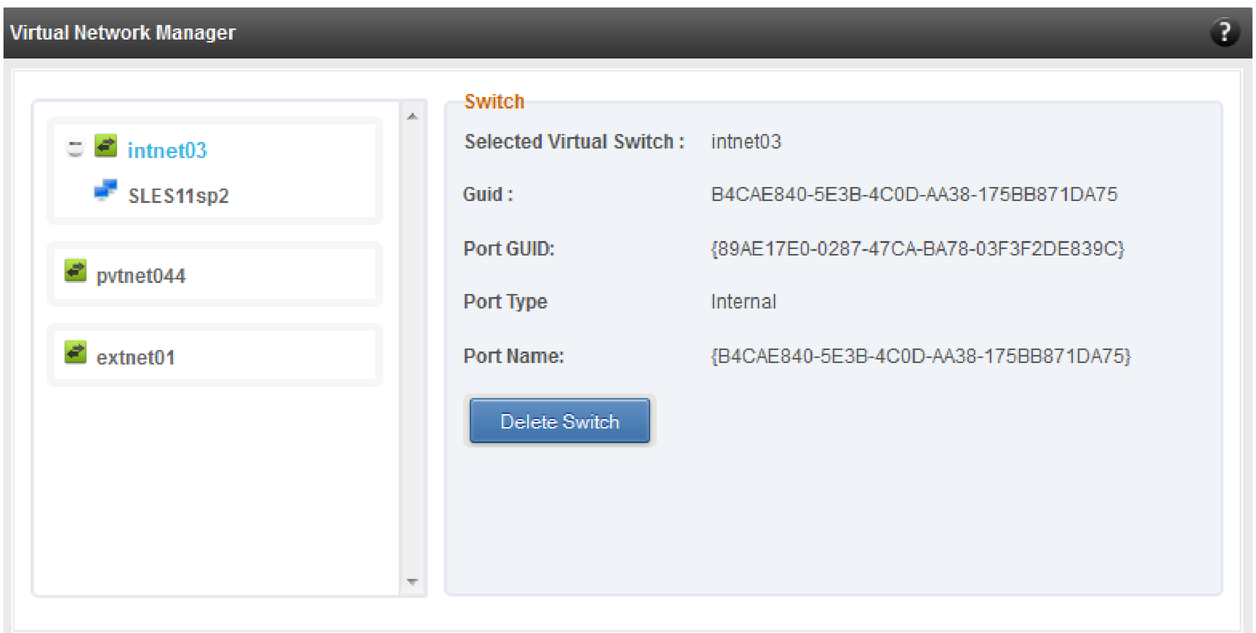


Figure 7.6.3.1 (h) – Virtual Network Manager module

Add Virtual Network Configuration (Windows)

There are three kinds of virtual networks you can create using this module:

- External network: Using this type, you can provide virtual machines access to external networks and vice versa via a physical network adapter in the host system. The virtual machines can also communicate with each other on the same virtual network.
- Internal Network: This type allows communication between virtual machines in the same virtual network and also between the virtual machines and the host. This type of virtual network does is not bound to any physical network adapter and no access to external networks is provided.

- Private Network: A Private Network is similar to Internal Network in that physical adapter is not required for setup and access to external networks is not provided. However, unlike Internal Network, guest operating systems can only communicate with guest operating systems in the same private network and not with the host. The host operating system cannot access the virtual machines on private network.

Once created, you can manage the virtual networks in the **Virtual Network Manager** module.

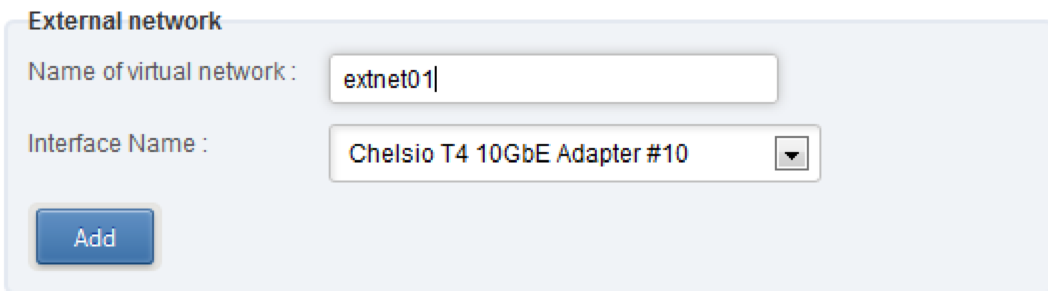


Figure 7.6.3.1 (i) – Creating external virtual network

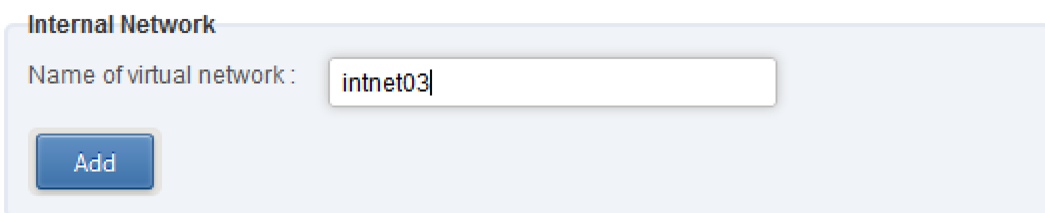


Figure 7.6.3.1 (j) – Creating internal virtual network

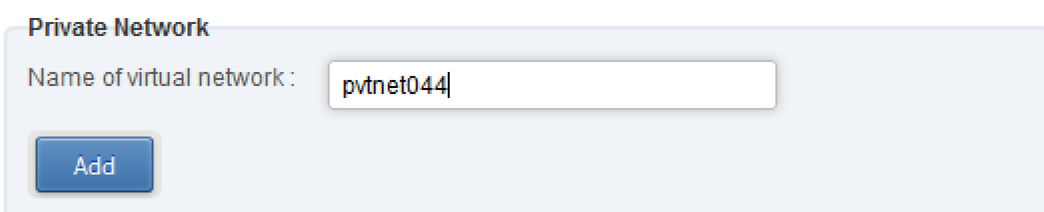


Figure 7.6.3.1 (k) – Creating private virtual network

Virtual Switch Settings (Windows)

To attach a virtual network to a virtual machine, select the virtual network from the **Virtual Network** list and the virtual machine from the **VM** list. Finally click *Attach*.

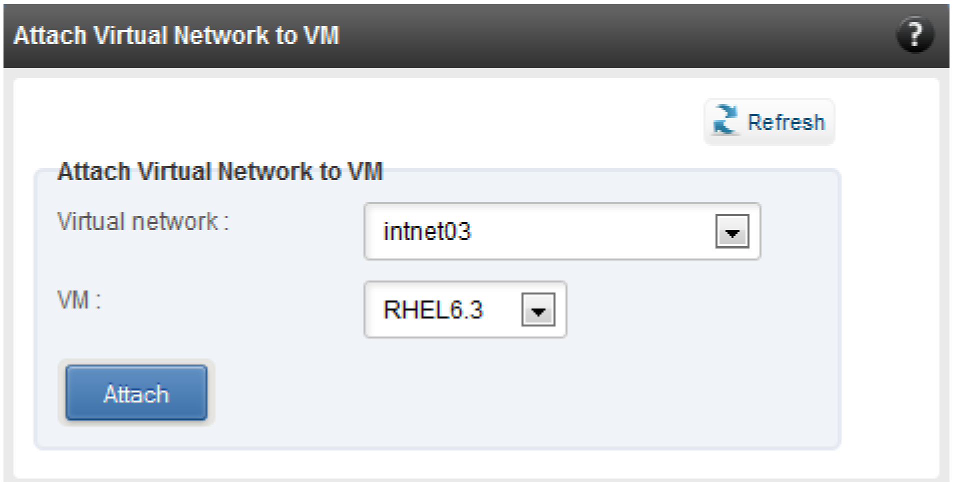


Figure 7.6.3.1 (l) – Attaching virtual network to VM (Windows)

7.6.4. iWARP

iWARP Settings

On Linux Agents, iWARP parameter settings for Chelsio's RDMA capable NICs can be set using this module. These settings can be set only when iWARP driver (*iw_cxgb4* for T4 and T5; *iw_cxgb3* for T3) is loaded. If you set any parameter for a T5 adapter, it applies for all the T5 adapters present. Same applies for T4 and T3 adapters.

On Windows Agents, only T3 HBAs are supported currently. Parameters can be set per port.

On FreeBSD Agents, only T4 CNAs are supported. iWARP parameter settings can be set only when *iw_cxgbe* driver is loaded.

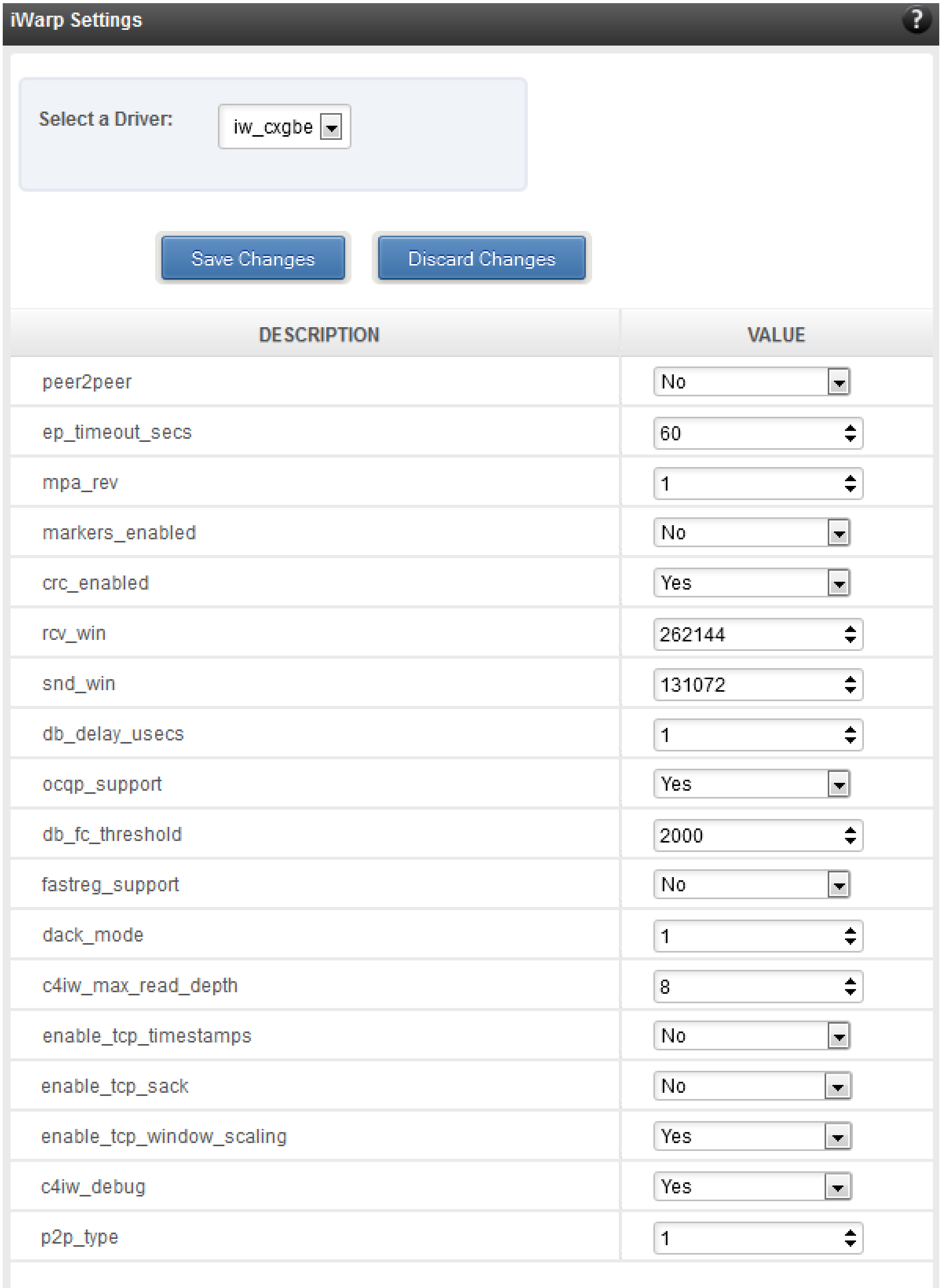


Figure 7.6.4 (a) – iWARP settings for T4 CNA for FreeBSD Agent

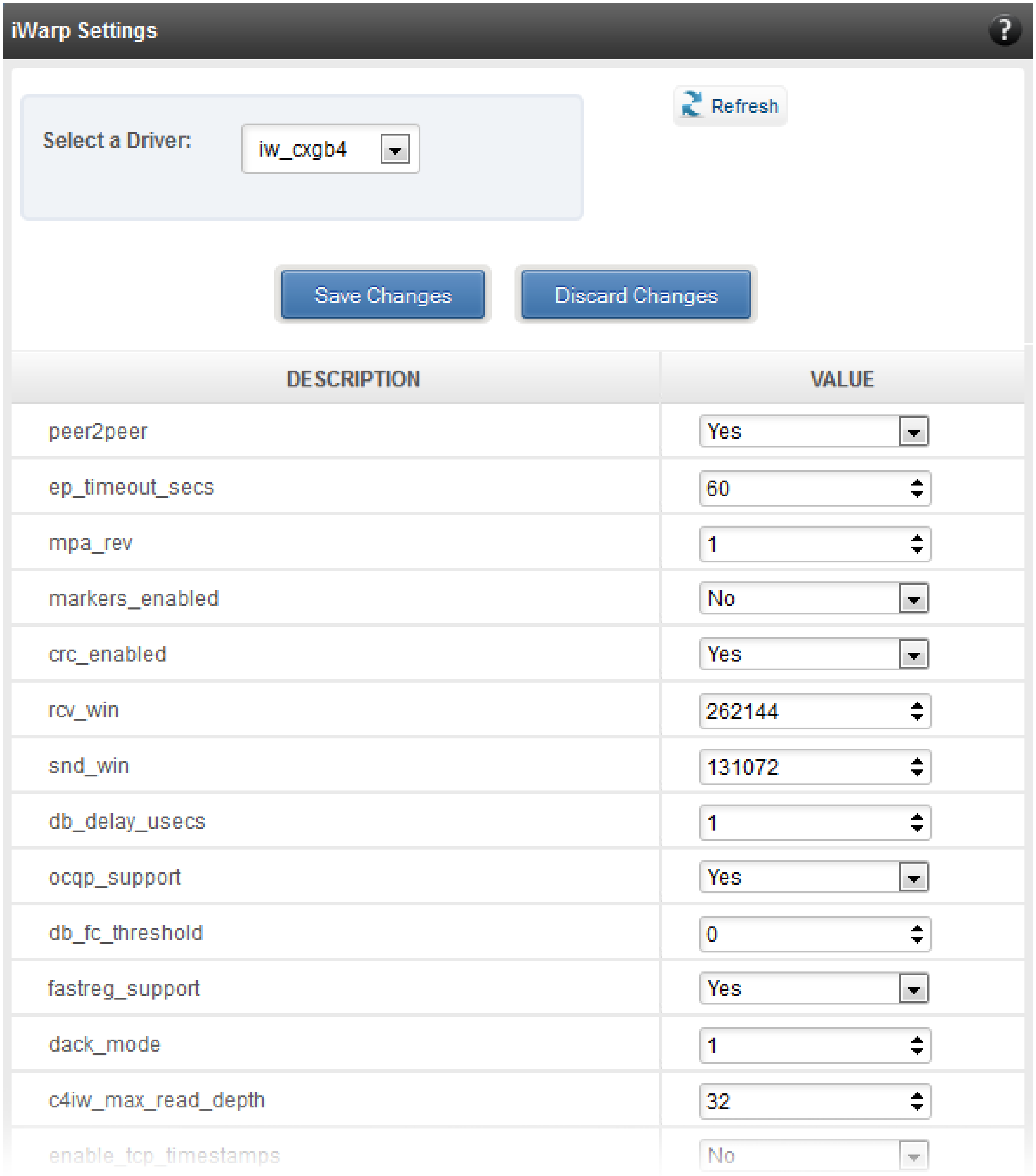


Figure 7.6.4 (b) – iWARP settings for T4 CNA for Linux Agent

7.6.5. Wire Direct

- **WD-UDP Process Statistics & Attributes**

The WD-UDP module lists the process ids (pid) of UDP traffic running on the agent and displays the corresponding statistics and attributes.

Note *Please ensure that WD-UDP traffic is running on the agent before accessing this module.*

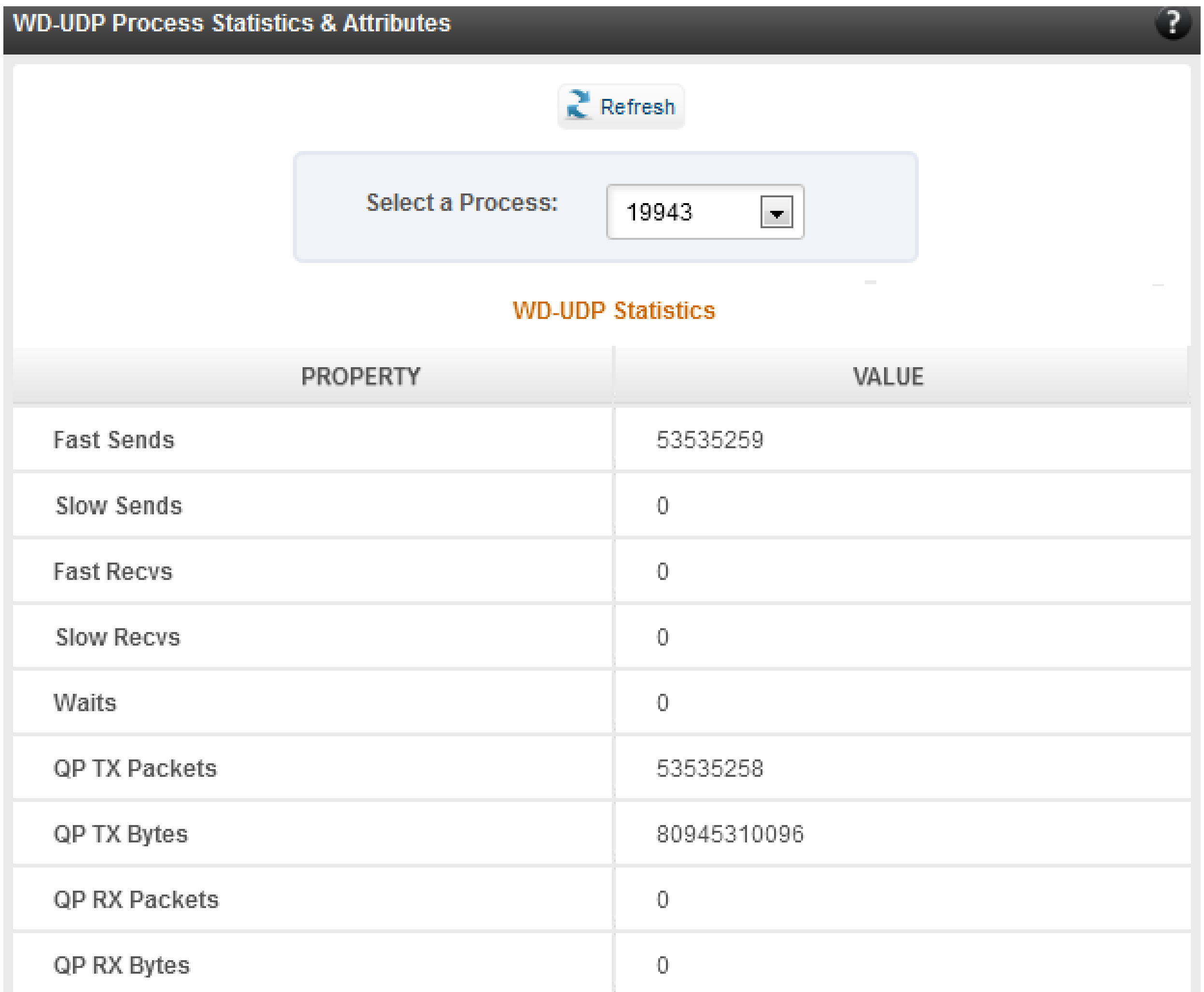


Figure 7.6.5 (a) – WD-UDP Process Statistics

WD-UDP Attributes

PROPERTY	VALUE
QP Number	51
Sockfd	5
State	BOUND
Device	eth8
Device Address	102.33.33.88
Filter ID	0
Local Address	102.33.33.88:40174
Remote Address	0.0.0.0:0
VLAN	-1
Priority	-1

Figure 7.6.5 (b) – WD-UDP Process Attributes

WD-TOE Process Statistics & Attributes

The WD-TOE module lists the process ids (pid) of TOE traffic running on the agent and displays the corresponding statistics and attributes.

Note *Please ensure that WD-TOE traffic is running on the agent before accessing this module.*

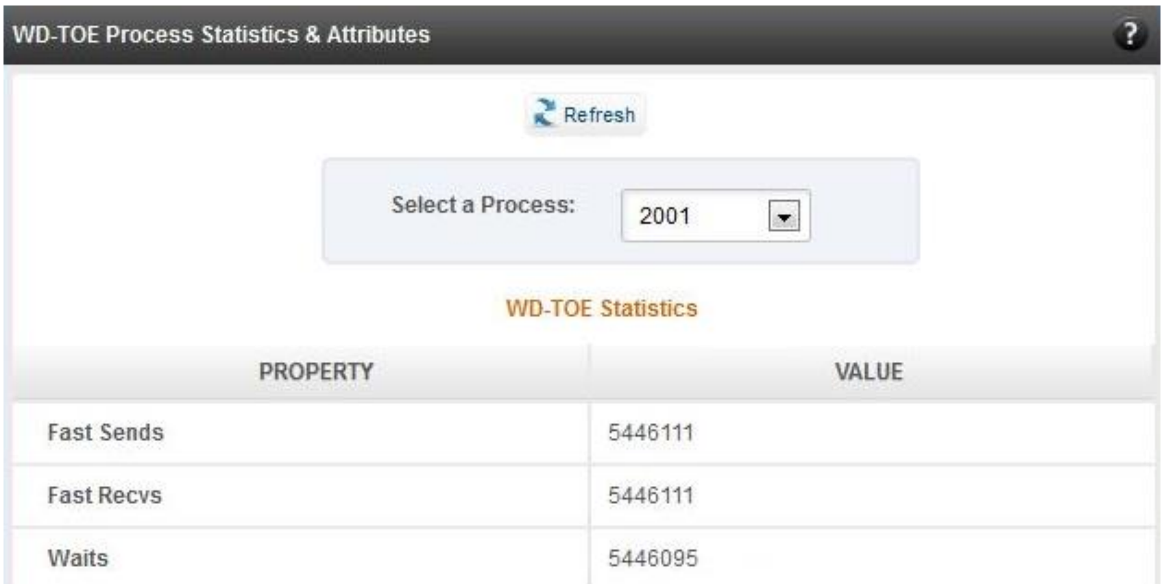


Figure 7.6.5 (c) – WD-TOE Process Statistics

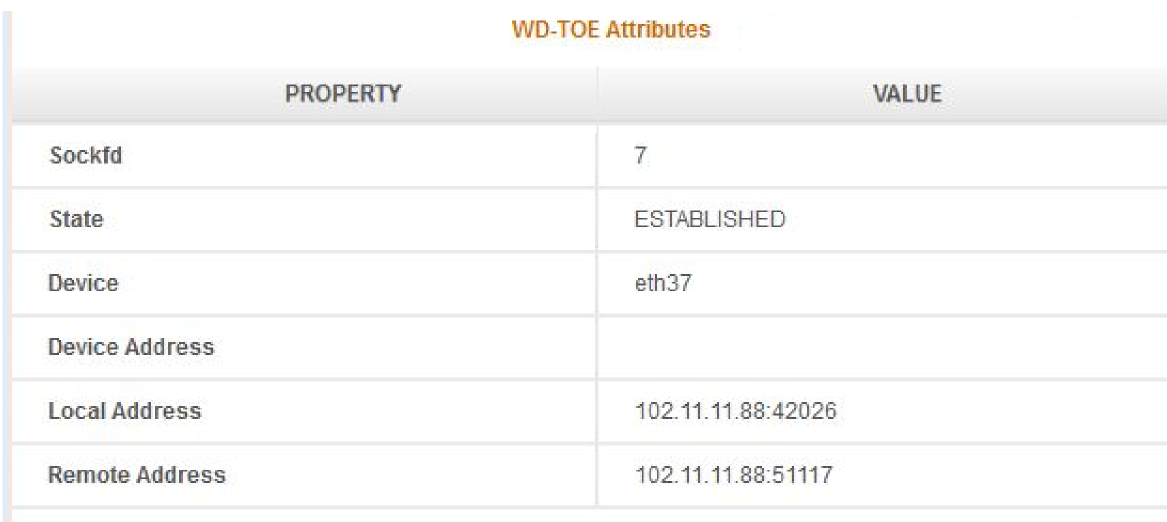


Figure 7.6.5 (d) – WD-TOE Process Attributes

7.7. Storage Page

Storage Summary

The **Storage** module lists the status of configuration modules under Storage section, running on the agent.

Storage

PROPERTY	VALUE
FCoE service on server	Enabled
iSCSI Initiator service on server	Enabled
iSCSI Target service on server	Enabled

Figure 7.7 – Storage Summary Module

7.7.1. FCoE Initiator (Linux, Windows, XenServer)

All supported Chelsio FCoE initiators available on the operating system can be managed from this page. FCoE support is extended on Linux, Windows and XenServer platforms. Please refer to **Platform/Driver matrix** section on the list of operating systems that are supported.

FCoE Initiator Summary

This module provides details about the driver installed; such as driver name and its version. The module also gives information about the number of FCoE enabled cards that are present on the machine.

FCoE Initiator Summary

PROPERTY	VALUE
FCoE Driver	csiostor (1.1.0.9)
No. of FCoE enabled cards	2
No. of FCoE Ports	6

Figure 7.7.1 (a) – FCoE Initiator Summary module for Linux Agent

7.7.1.1. FCoE Initiator Card

FCoE Card Summary

Details pertaining to the card used such as model, firmware/hardware version etc, are provided in this module.

FCoE Card Summary

PROPERTY	VALUE
Vendor ID	1425
Card Serial Number	PT41110672
Number of FCoE Ports	4
Manufacturer	Chelsio T440-LP-CR 10G [FCoE]
Model	T440-LP-CR
Hardware Version	T440-LP-CR 10G
Firmware Version	1.7.0.0

Figure 7.7.1.1 (a) – FCoE Card Summary module

FCoE Attributes

Information such as Interrupt modes (MSI/MSI-X/INTx), SCSI mode and the card state are provided in this module.

FCoE Attributes

PROPERTY	VALUE
Interrupt Mode	MSI-X
SCSI Mode	Initiator
State	READY

Figure 7.7.1.1 (b) – FCoE Attributes module

7.7.1.2. FCoE Port

This is an actual N_Port which communicates with the fabric and performs FIP and FCoE device discovery. This page lets the user to retrieve all the FCoE specific port information and also extend NPIV management support. It contains the following sections:

- **FCoE Port Summary**

The SCSI adapter name and the underlying ENODE MAC address of the physical port can be found here.

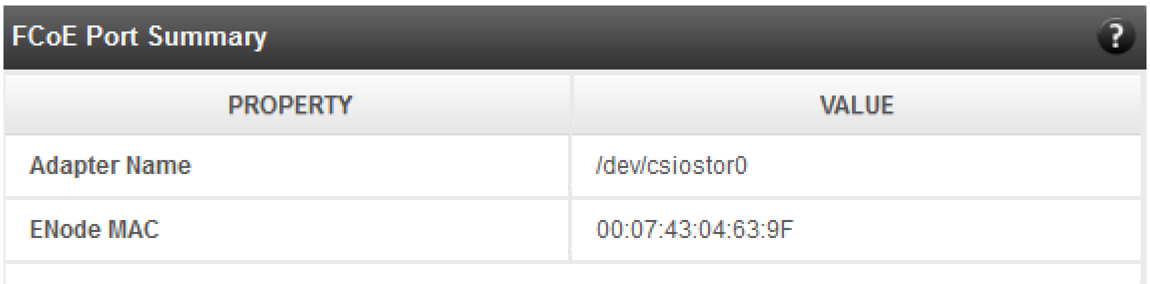


Figure 7.7.1.2 (a) – FCoE Port Summary module for Linux Agent

FCoE Port Attributes

This module provides details about link status and port identifiers such as WWPN, WWNN, FC ID and NPort MAC Address. The module also contains fabric information such as fabric name, VLAN on which the FCoE service is currently running and the number of SCSI targets that are being discovered by this port. Port speed being mentioned in this section varies on the card type (10G/1G) being used. Note that only class 3 service is supported by the initiator for now and the frame size is fixed to 2128 bytes as per spec.

FCoE Port Attributes

PROPERTY	VALUE
State	Operational
NodeWWN	50:00:74:30:46:39:F0:00
PortWWN	50:00:74:30:46:39:F0:80
NPort MAC Address	0E:FC:03:53:00:23
Vlan ID	2
Fabric Name	20:02:00:05:73:D5:7A:C1
NPort ID	53:00:23
Type	NPort
Supported Class of Service	3
OS Device Name	/sys/class/fc_host/host119
Speed	10 GBPS
Maximum Frame Size	2128
No. of SCSI Targets	0

Figure 7.7.1.2 (b) – FCoE Port Attributes module for Linux Agent

FCoE NPIV management

NPIV is a fibre channel facility allowing multiple N_Port IDs to share a single physical N_Port. This module allows the user to manage virtual ports on the corresponding FCoE Port.

To create a virtual port, select the option **Create** and the GUI allows two ways of creating a virtual port.

i. Manual: Where the user can manually create a virtual port by providing a value to the WWPN and WWNN fields.
ii. Auto-generate: Where the FCoE function auto-generates a WWPN and WWNN for the virtual port.

To delete a virtual port, select the option **Delete** and select the virtual port WWPN which you want to delete and click on **delete**.

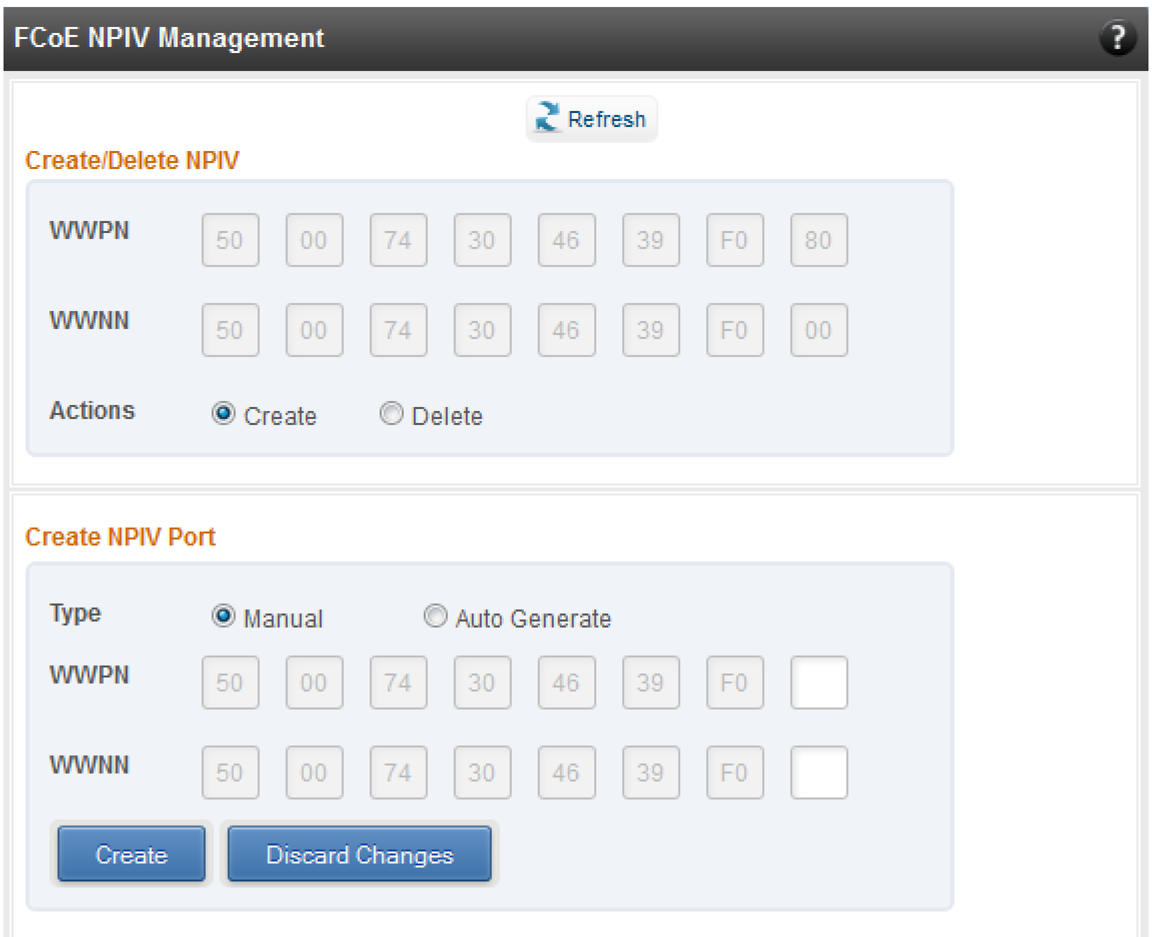


Figure 7.7.1.2 (c) – FCoE NPIV management module

7.7.1.3. FCoE Remote Port

Remote ports are the SCSI targets that are discovered by their respective N_port/virtual ports. The GUI conveys the same via a tree structure so that the end user knows the initiator-target mapping.

- **FCoE Remote Port Attributes**

This module provides details about the discovered target such as target's FC ID, WWPN and WWNN so that the user can identify the discovered target accordingly.

FCoE Remote Port Attributes

PROPERTY	VALUE
FC ID	54:00:53
State	Operational
NodeWWN	20:01:00:11:0D:56:29:00
PortWWN	20:01:00:11:0D:56:29:00

Figure 7.7.1.3 (a) – FCoE Remote Port Attributes module

FCoE Remote Port Lun Details

This module provides the LUN information such as size of the LUN, SCSI address, and LUN address. For Linux, the SCSI address is displayed in H:C:T:L (Host:Channel:Target:Lun) format and for Windows, it is displayed in P:B:T:L(SCSI Port:Bus:Target:Lun) format.

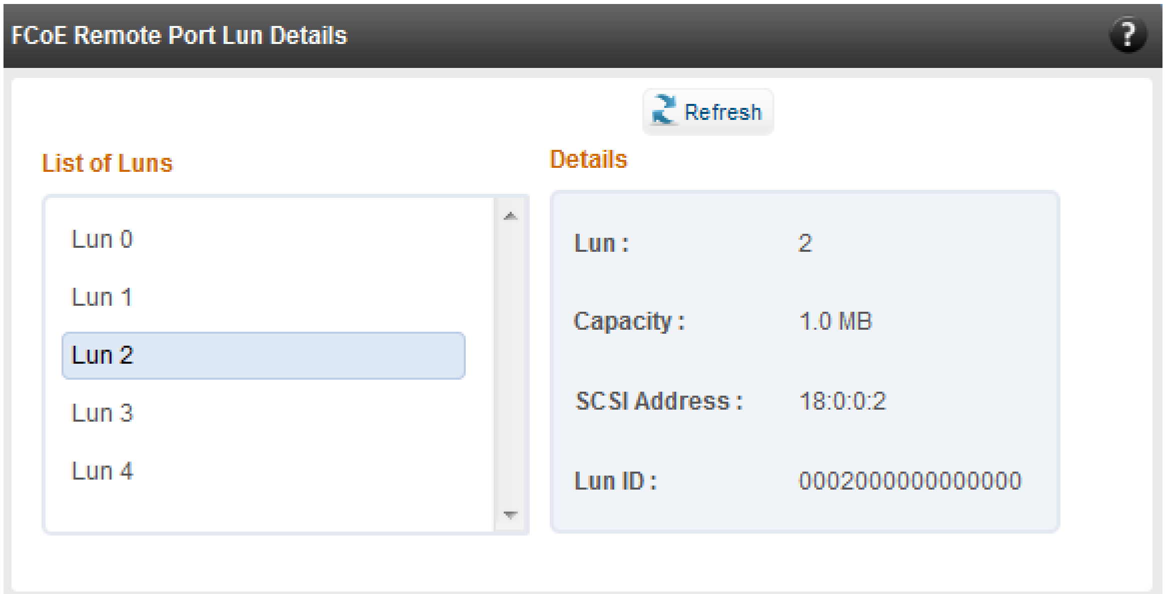


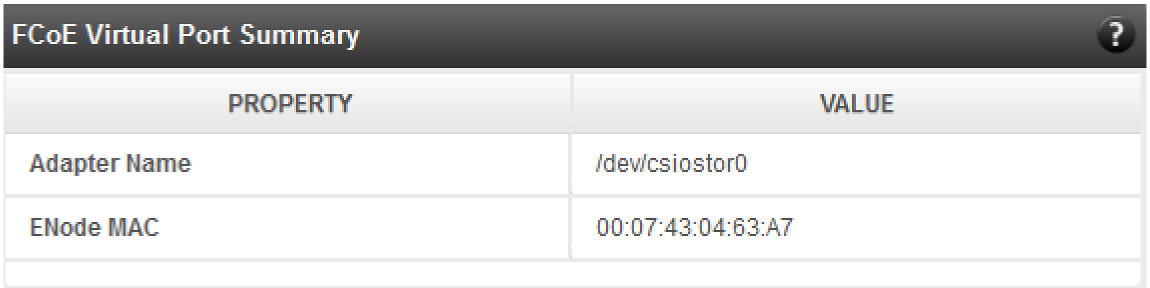
Figure 7.7.1.3 (b) – FCoE Remote Port Lun Details module

7.7.1.4. FCoE Virtual Port

A virtual port allows multiple Fibre Channel initiators to occupy a single physical port, easing hardware requirements in SAN design, especially where virtual SANs are called for. The virtual ports appear under their respective N_Ports after creation and the GUI conveys it via a tree structure so that the end user knows the N_port-VN_Port mapping. It contains the following modules:

- #### FCoE Virtual Port Summary

The SCSI adapter name and the underlying ENODE MAC address of the physical port can be found here.


FCoE Virtual Port Summary

PROPERTY	VALUE
Adapter Name	/dev/csiostor0
ENode MAC	00:07:43:04:63:A7

Figure 7.7.1.4 (a) – FCoE Virtual Port Summary module for Linux Agent

FCoE Virtual Port Attributes

The module provides details about link status and port identifiers such as WWPN, WWNN, FC ID and Virtual NPort MAC Address. The module also contains fabric information such as fabric name, VLAN on which the FCoE service is currently running and the number of SCSI targets that are being discovered by this virtual port. Port speed being mentioned in this section varies on the card type (10G/1G) being used. Note that only class 3 service is supported by the initiator for now and the frame size is fixed to 2128 bytes as per spec.

FCoE Virtual Port Attributes

PROPERTY	VALUE
State	Operational
NodeWWN	50:00:74:30:46:3A:71:09
PortWWN	50:00:74:30:46:3A:71:89
NPort MAC Address	0E:FC:03:77:00:1D
Vlan Id	5
Fabric Name	20:05:00:05:73:D5:7A:C1
Nport ID	77:00:1D
Type	VN_Port
Supported Class Of Service	3
OS Device Name	/sys/class/fc_host/host127
Speed	10 GBPS
Maximum Frame Size	2128
No. of SCSI Targets	1

Figure 7.7.1.4 (b) – FCoE Virtual Port Attributes module

FCoE Remote Port Attributes

This module provides details about the discovered target for remote port associated with virtual port. Details such as target's FC ID, WWPN and WWNN are provided so that the user can identify the discovered target accordingly.

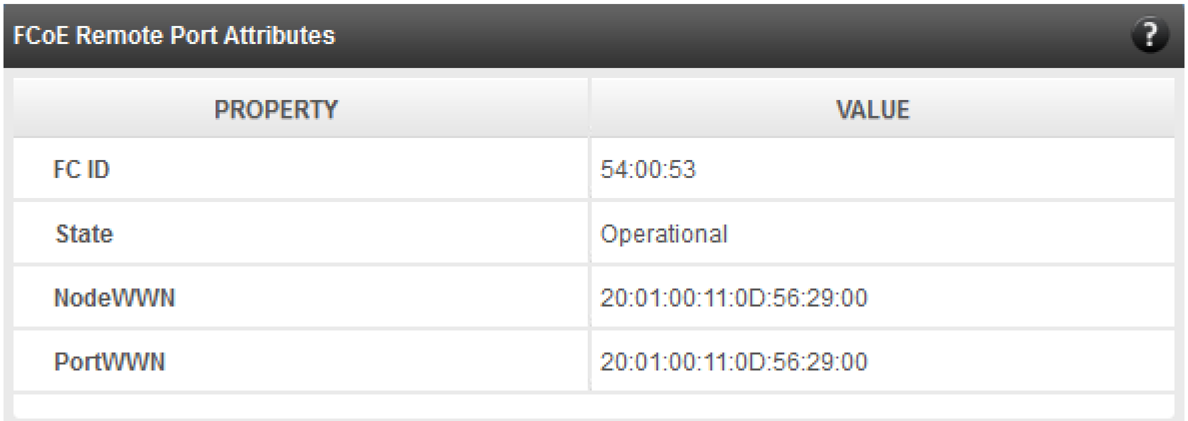


Figure 7.7.1.4 (c) - FCoE Remort Port Attributes module

FCoE Remote Port Lun Details

This module provides LUN information for remote port associate with virtual port. Details such as size of the LUN, SCSI address, and LUN address are provided. For Linux, the SCSI address is displayed in H:C:T:L (Host:Channel:Target:Lun) format and for Windows, it is displayed in P:B:T:L(SCSI Port:Bus:Target:Lun) format.

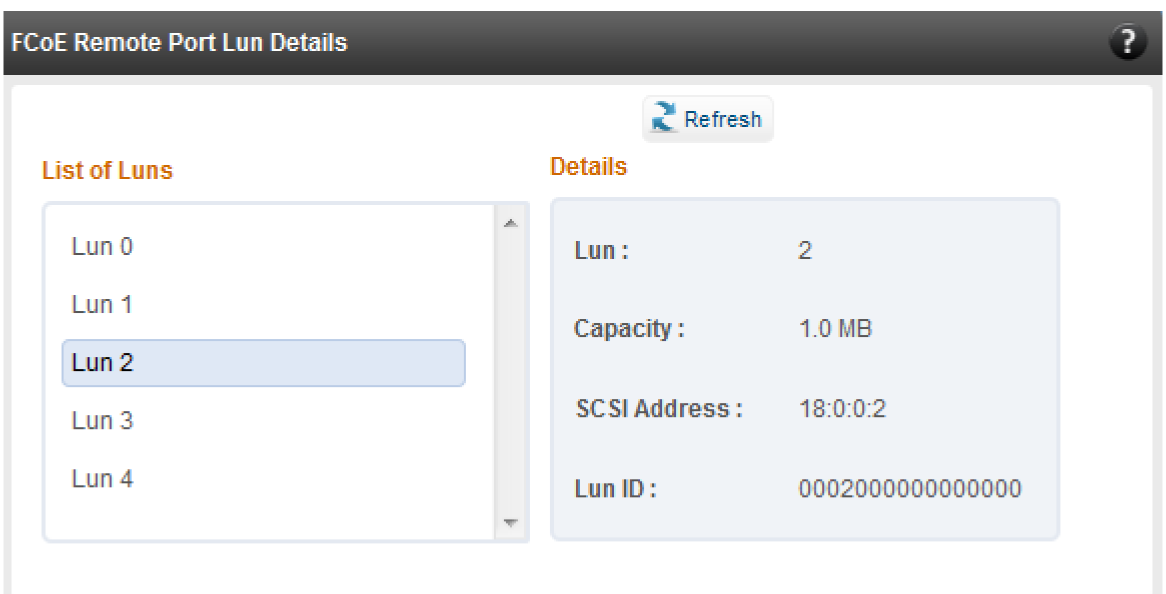


Figure 7.7.1.4 (d) - FCoE Remote Port Lun Details module

7.7.2. iSCSI initiator (Linux, Windows)

All supported iSCSI initiators can be managed from this page. The supported initiators on Windows are Microsoft and Chelsio iSCSI initiator (T5/T4 adapters). On Linux, Open iSCSI initiator is supported. The modules available on this page are:

- **Initiator nodes**

This module lists the initiator nodes / virtual adapters configured in the initiator stack. The node can be enabled or disabled (Chelsio node cannot be disabled in Windows), and its properties can be viewed and edited in this module. In the Chelsio Linux stack, new initiator nodes can be created too. Disabling the initiator causes it to log out of any iSCSI targets that it is connected to, thus removing any disks provided by the iSCSI targets that were connected. Use the **Disable** option with caution. The CHAP authentication secret should be between 12 and 16 characters in length, and the initiator's IQN name should start with "iqn."

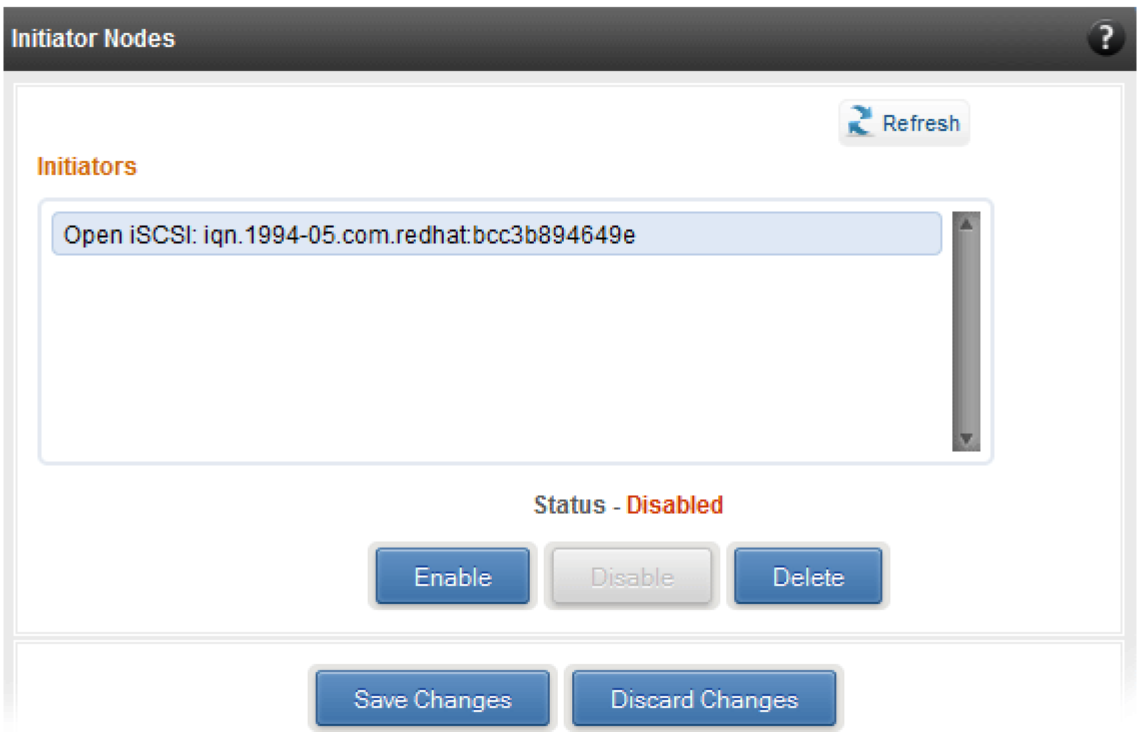


Figure 7.7.2 (a) - Open iSCSI initiator

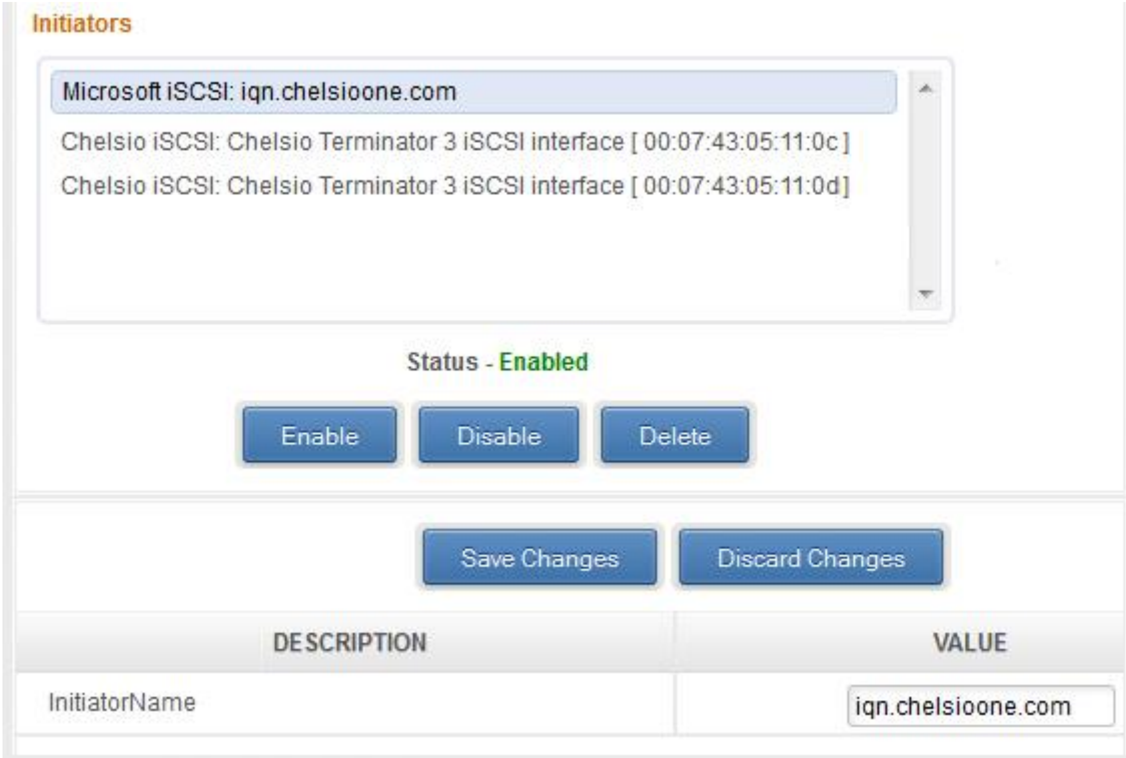


Figure 7.7.2 (b) - Microsoft iSCSI initiator

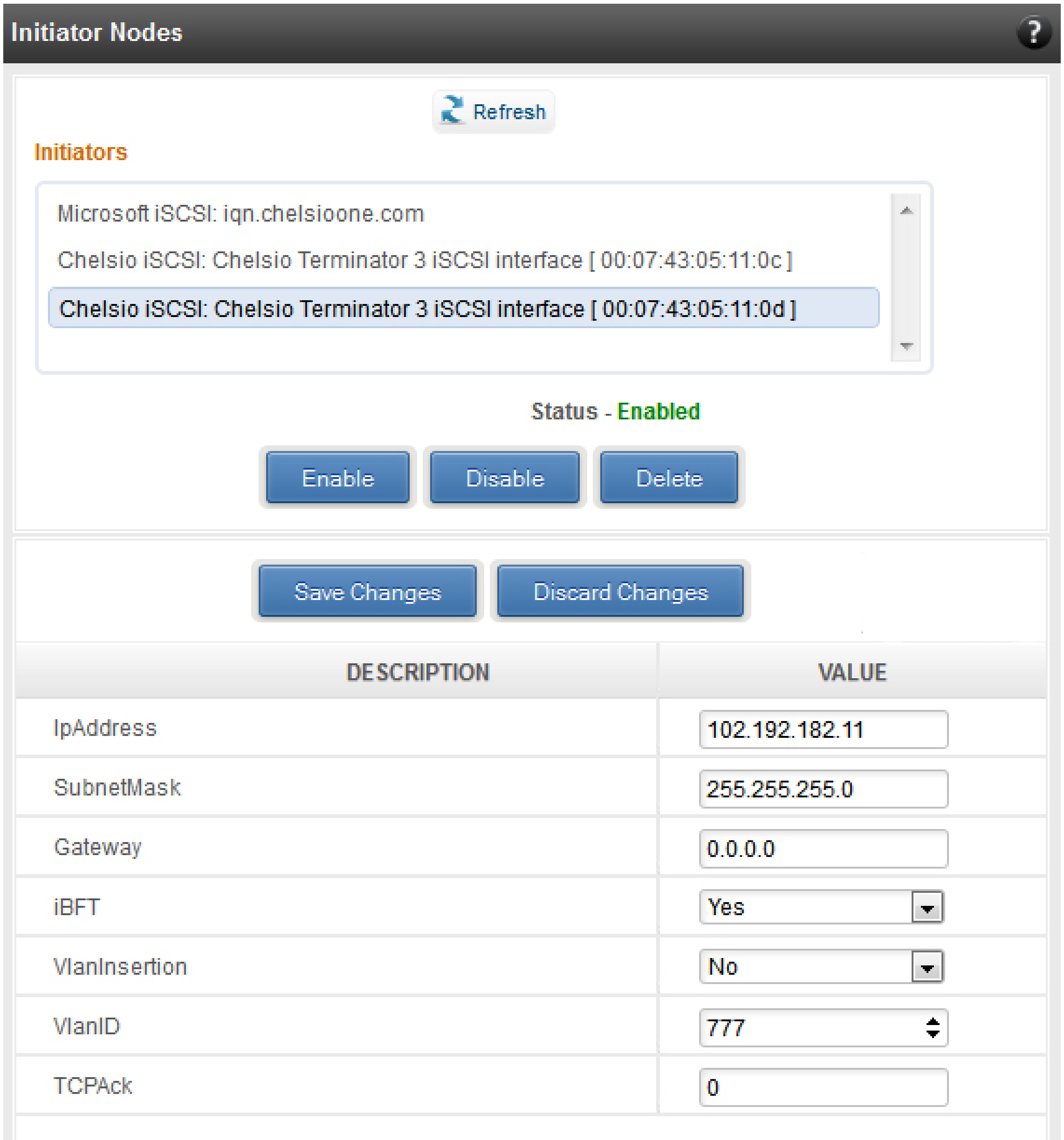


Figure 7.7.2 (c) - Chelsio iSCSI initiator

Discover targets

iSCSI targets can be discovered by providing the IP address and TCP port (usually 3260) of the target. In Windows, you can specify the initiator HBA to use and its IP address. The discovery operation fetches the targets found at that Portal (combination of IP address and TCP port). The discovery operation also fetches all the other Portals that the target(s) are listening on. The discovered target can be deleted if required. Please note that all the Portals that the target sent are listed. The delete operation will not work on all the portals, only on the original discovery portal (the IP address and TCP Port specified when discovering the target).

Note *If there are any pre-existing iSCSI sessions established to the target, deletion of the target Portal from the discovered targets list will fail.*

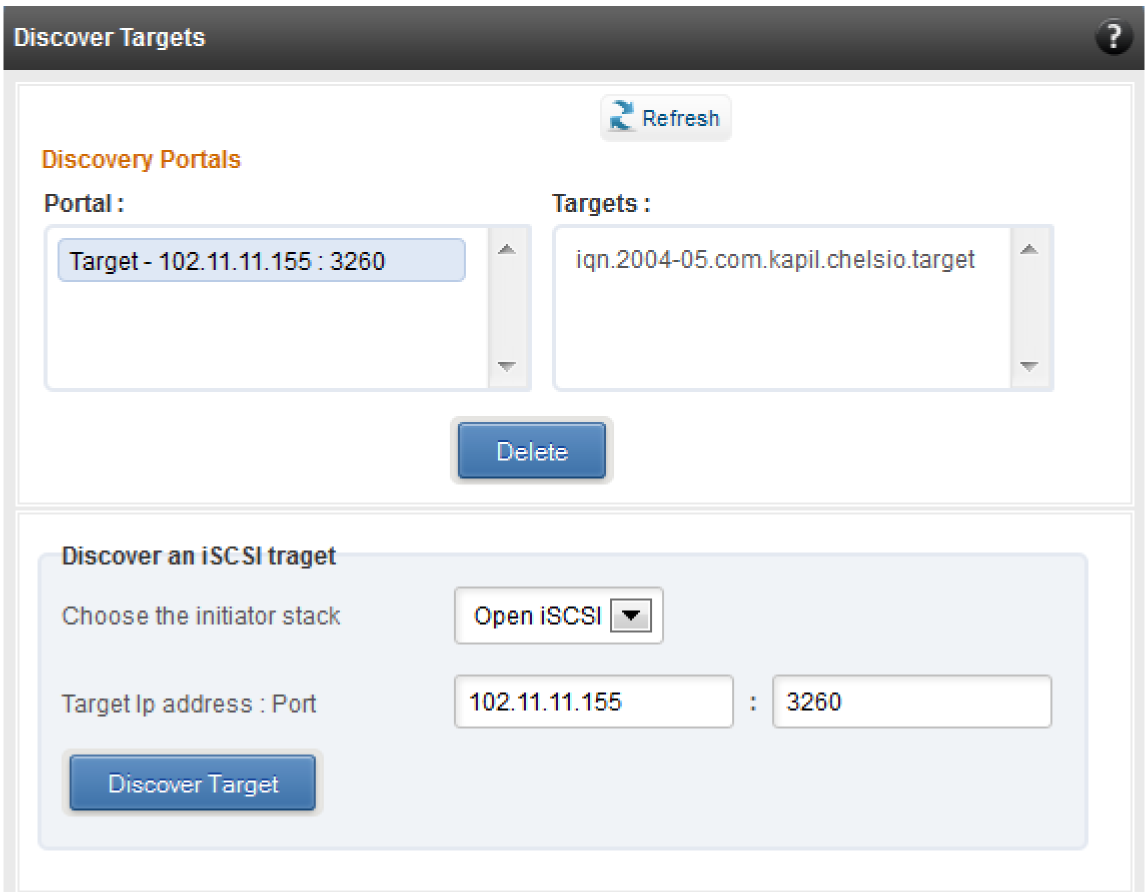


Figure 7.7.2 (d) - Discover targets module

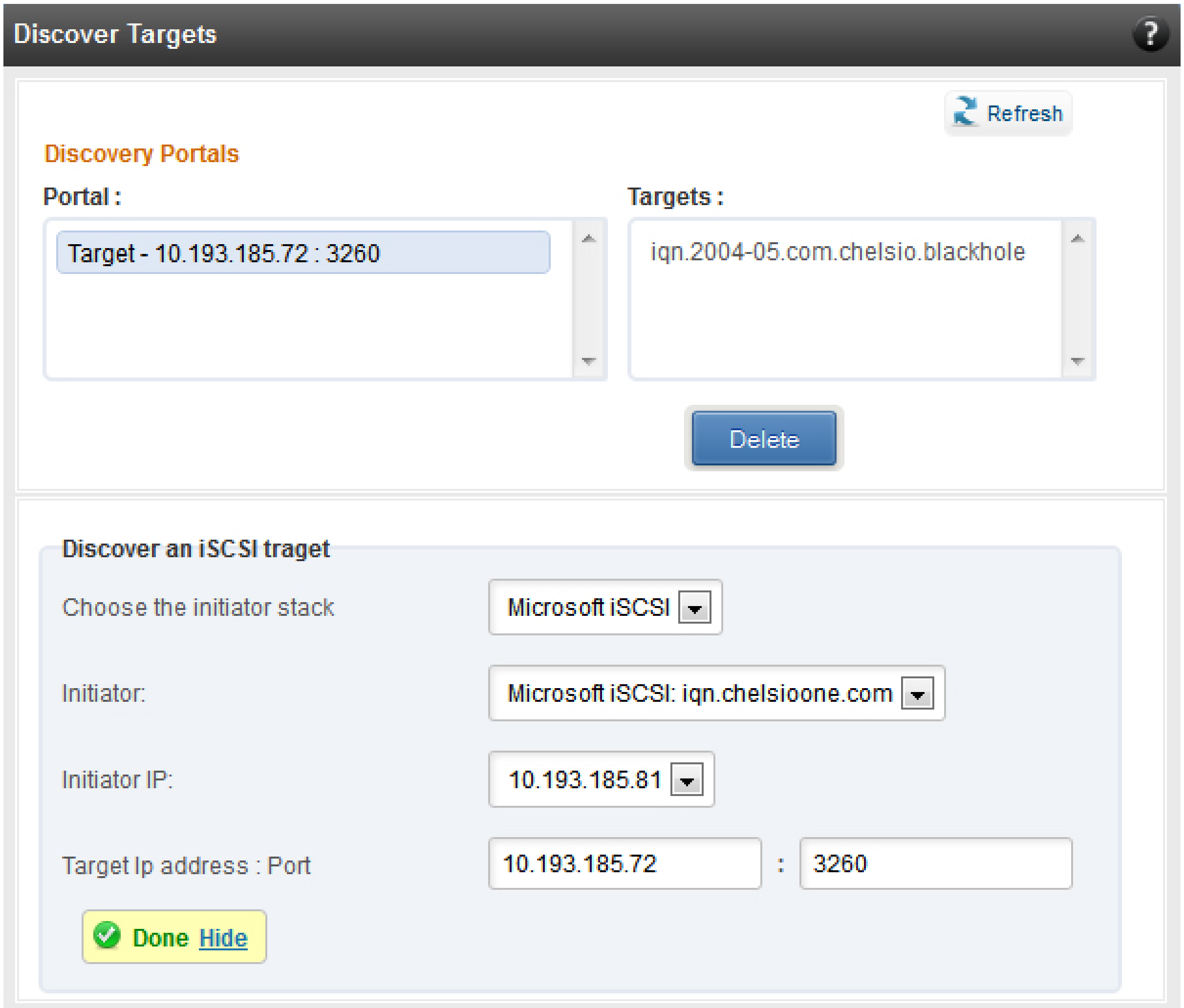


Figure 7.7.2 (e) - Discover targets module

Targets

The iSCSI targets that have been discovered, or are currently connected, are listed here. You may login, logout and delete the target from the initiator's configuration. In Windows, for the Microsoft iSCSI initiator, connections to an already established iSCSI session can be added or deleted. For the Microsoft iSCSI initiator or the Open iSCSI initiator, you may specify the authentication details and digest settings while logging in. For Chelsio Linux initiator, these settings should be set prior to attempting a login, in the Initiator nodes module. If a target is connected, the sessions and connections to the target, and the disks provided by the target will be listed.

In Windows, you can specify the initiator HBA to use and its IP address while logging in.

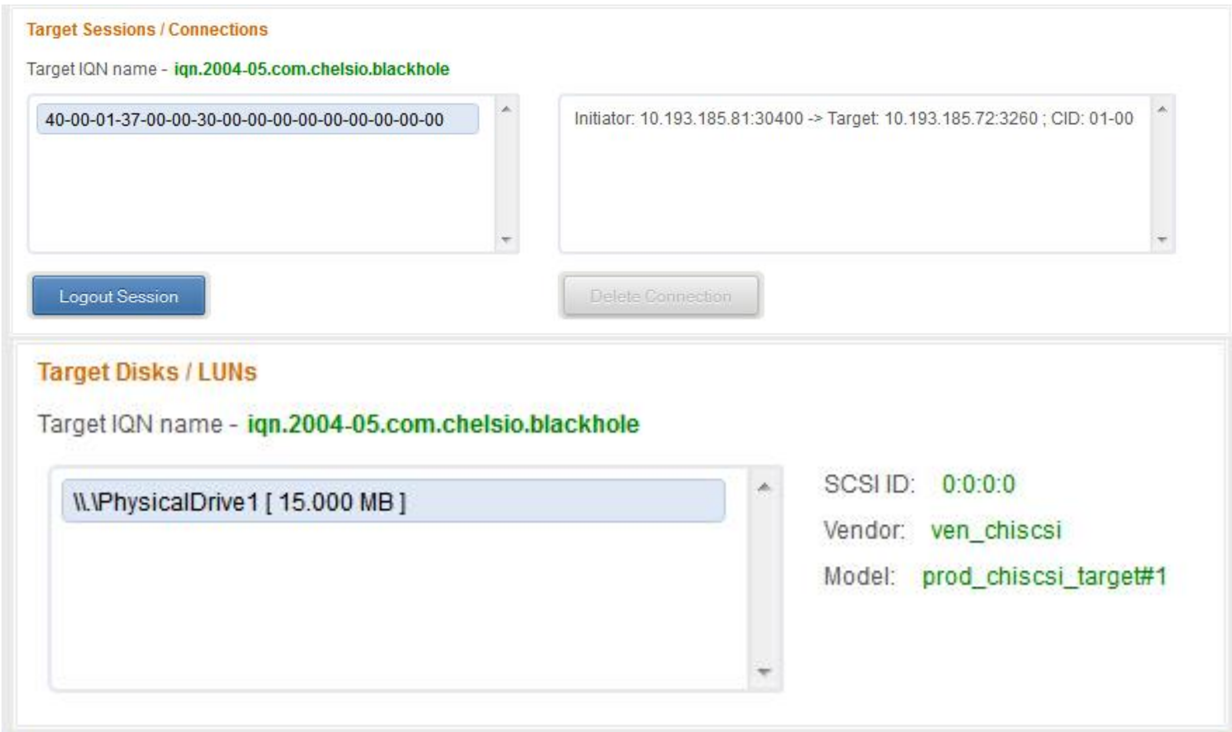


Figure 7.7.2 (f) - Targets module after logging in

7.7.3. FO iSCSI Initiator (Linux)

Full Offload iSCSI Hardware Information

PCI, firmware and other adapter related details are provided in this module. Select the Chelsio adapter for which you want to view properties from the **Select a T4 Card** drop-down list and the module will expand to display related properties. You can also view details like link id, status, enode mac, etc of all the ports of the selected adapter.

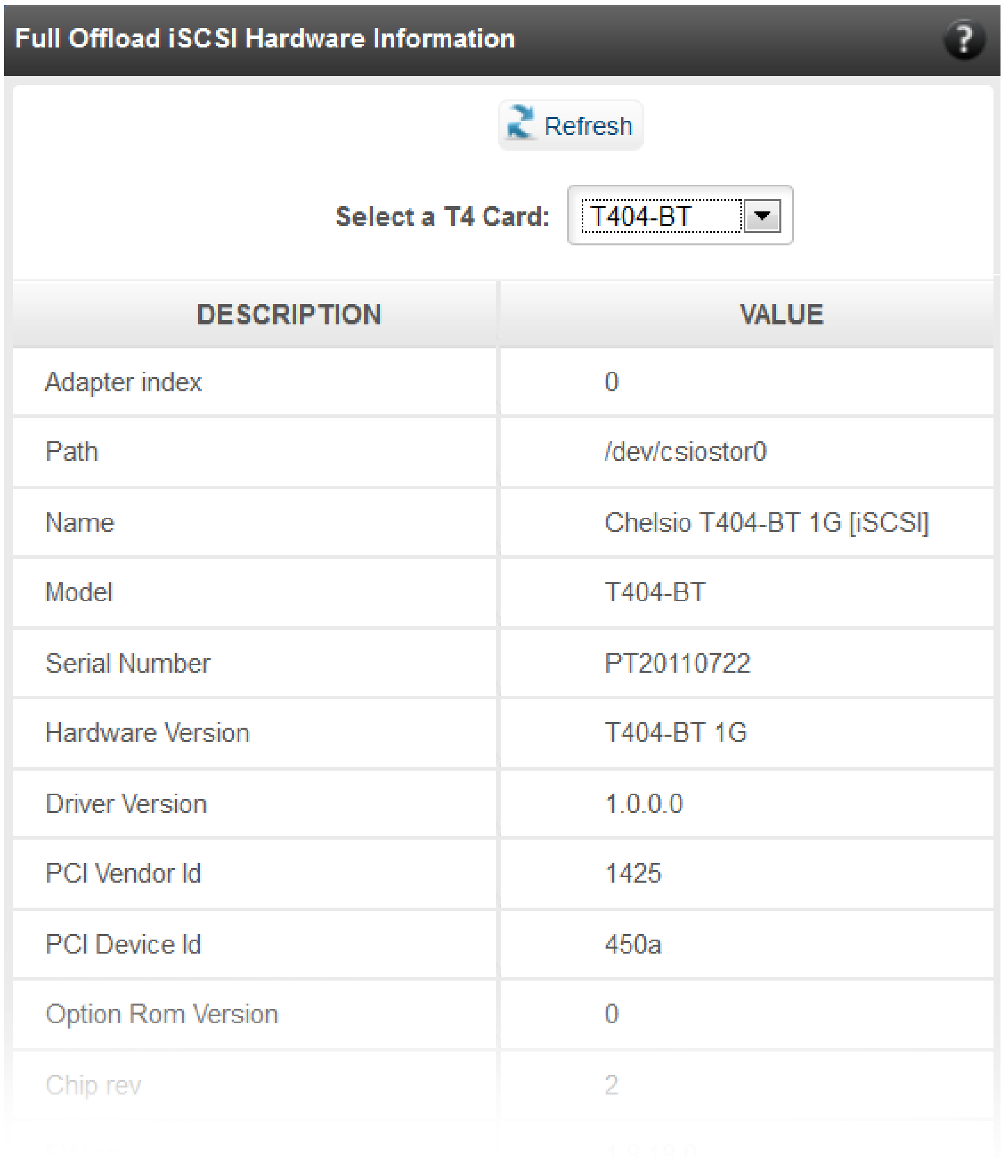


Figure 7.7.3 (a) - Full Offload iSCSI Hardware Information module

FO iSCSI Manage Ports

Here you can configure various port settings like VLAN id, Maximum Transmission Unit (MTU) and IP. Select a Chelsio adapter from **Select a T4 Card** drop-down list and then select the port for which you want set any of the aforementioned properties. MTU can be set between 1500-9000 bytes. VLAN id can be set within the range 0-4094 (enter 0 to disable it). The IP type can be *IPV4* (static) or *DHCP*.

The **Port Up** and **Port Down** buttons will enable and disable the selected port respectively. The **Clear IP** button deletes values set for the IP Type, IP, Subnet Mask and Gateway properties and resets them.

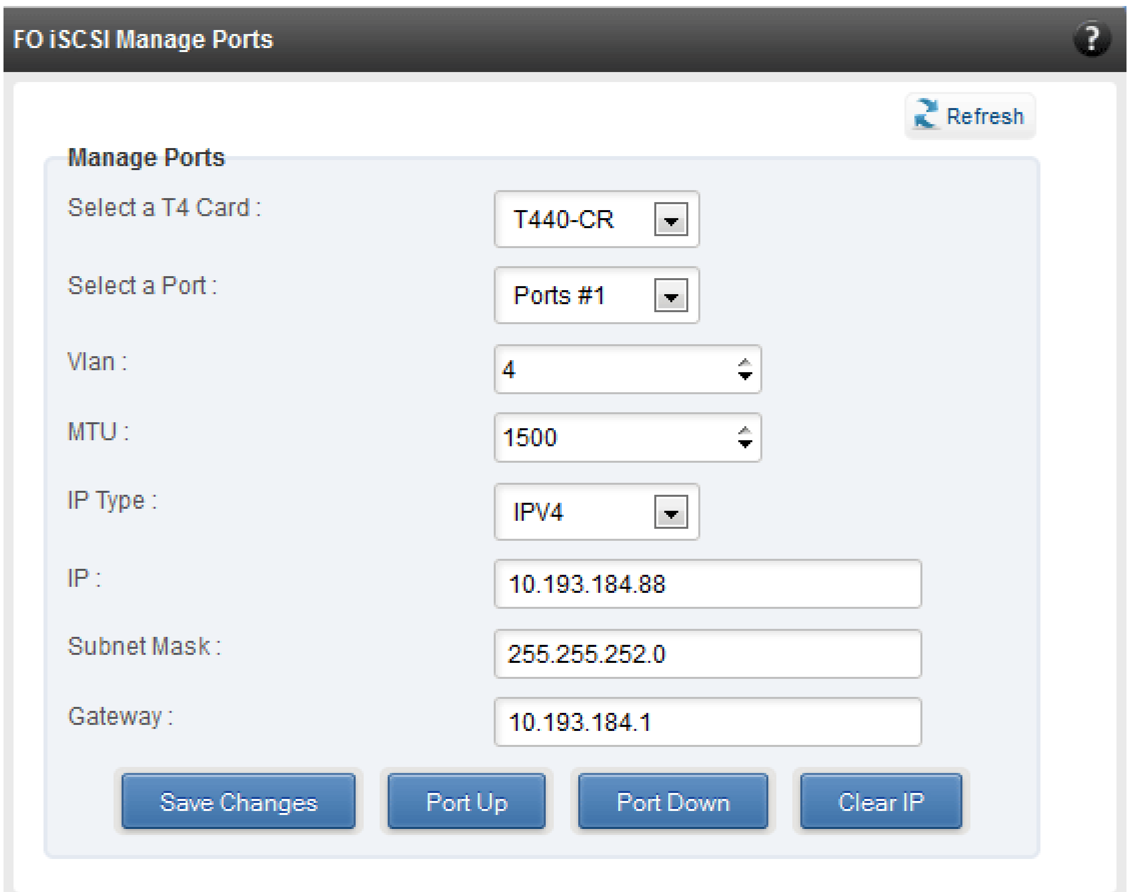


Figure 7.7.3 (b) - FO iSCSI Manage Ports module

FO iSCSI Initiator Properties

In the **FO iSCSI Initiator Properties** module, you can configure FO iSCSI Initiator by setting different properties like enabling/disabling CHAP authentication, setting Header and Data digest, etc.

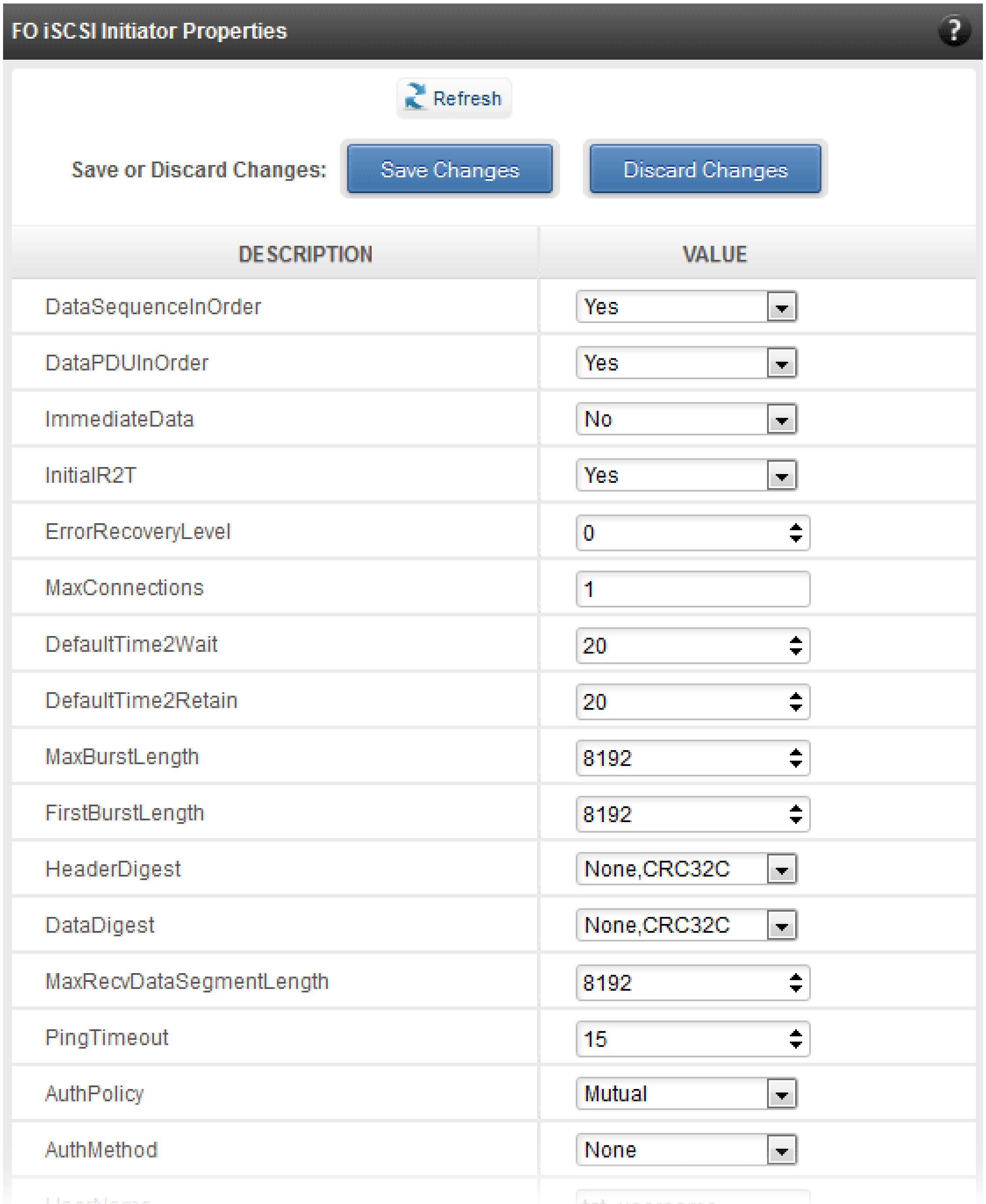


Figure 7.7.3 (c) - FO iSCSI Initiator Properties

FO iSCSI Manage Instances

The FO iSCSI Initiator service maintains multiple instances of a target depending on the discovery method. In this module, you can set upto 8 instances. Configurable parameters include initiator node name (IQN), alias (friendly) name, Initiator (CHAP) Username and password.

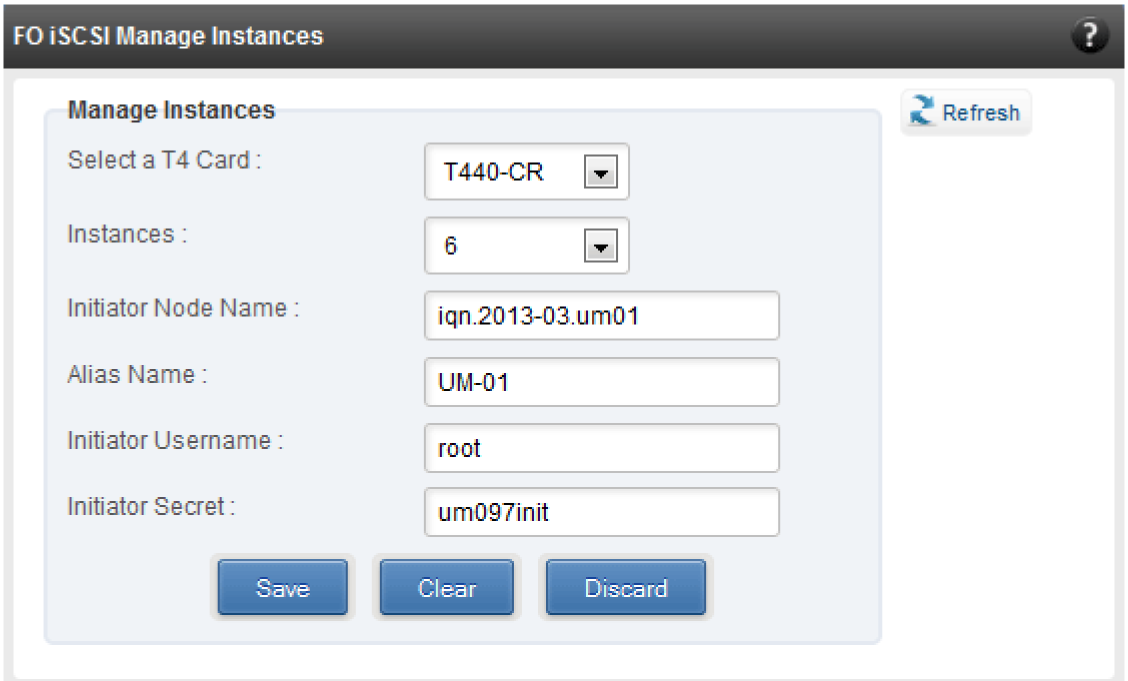


Figure 7.7.3 (d) - FO iSCSI Manage Instances module

FO iSCSI Discover Details

iSCSI Targets can be discovered using this module. Select a Chelsio adapter and initiator instance using which you want to discover targets. Next, provide the source (initiator) and destination (target) IP. Finally, click **Discover**. After successful discovery, all the discovered targets will appear in the **Discovered Targets** section. To view more details, click on the Target name.

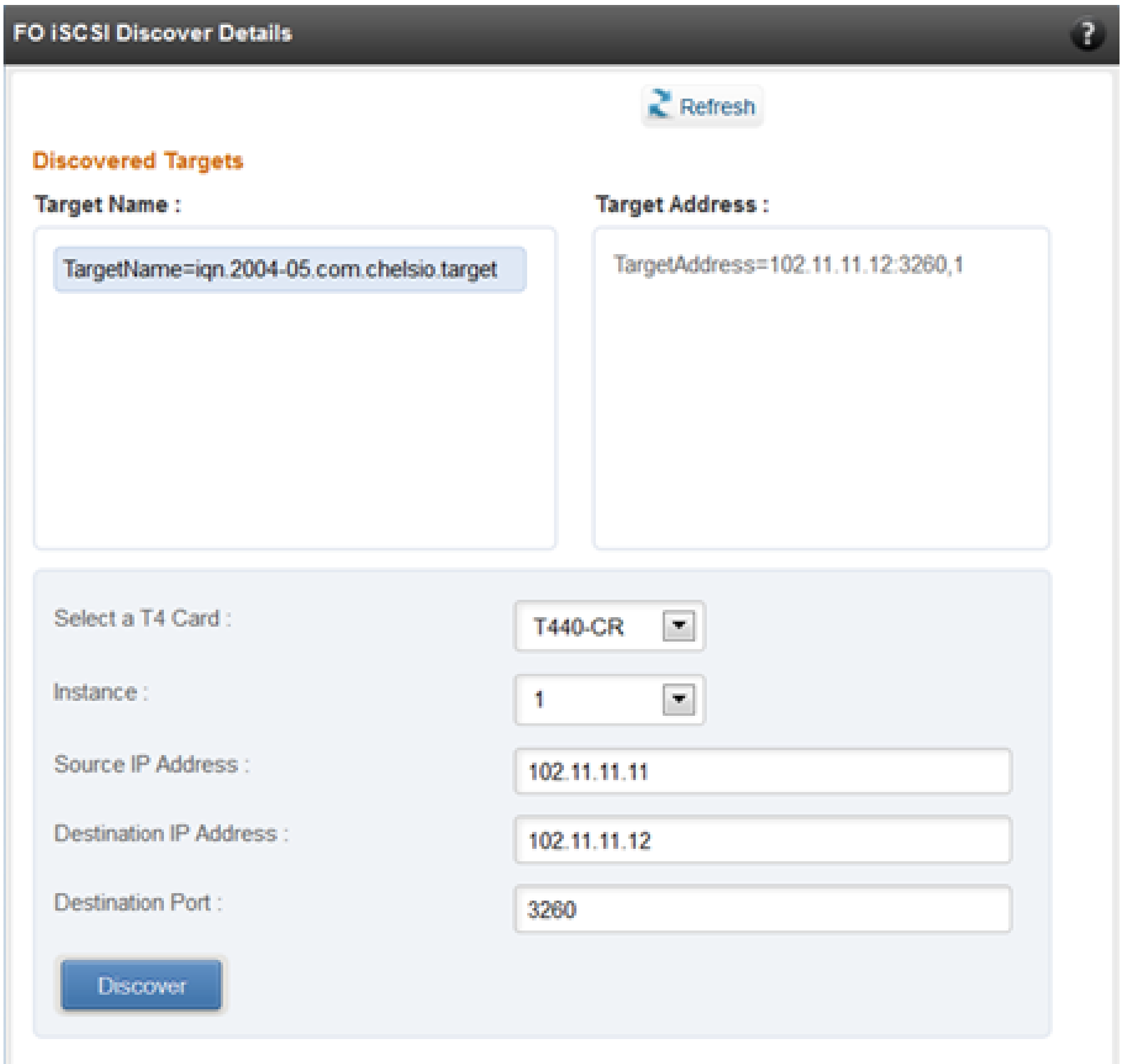


Figure 7.7.3 (e) - FO iSCSI Discover Details module

FO iSCSI Session Details

The FO iSCSI Session Details module can be used to log onto targets and view details of established iSCSI sessions. You can also logout from a target

Use the **Login** section to connect to a target. *adapter*, *(initiator) instance*, *Target Name*, *Source (Initiator) IP*, *Destination (Target)* IP and *Destination Port* are mandatory. After providing values for these fields, click **Login.**

By default, no authentication mechanism is used while connecting to a target. You can however configure CHAP for a secure iSCSI connection. **One-way** (target authenticates the initiator) and **Mutual** (target and initiator authenticate each other) authentication methods are supported.

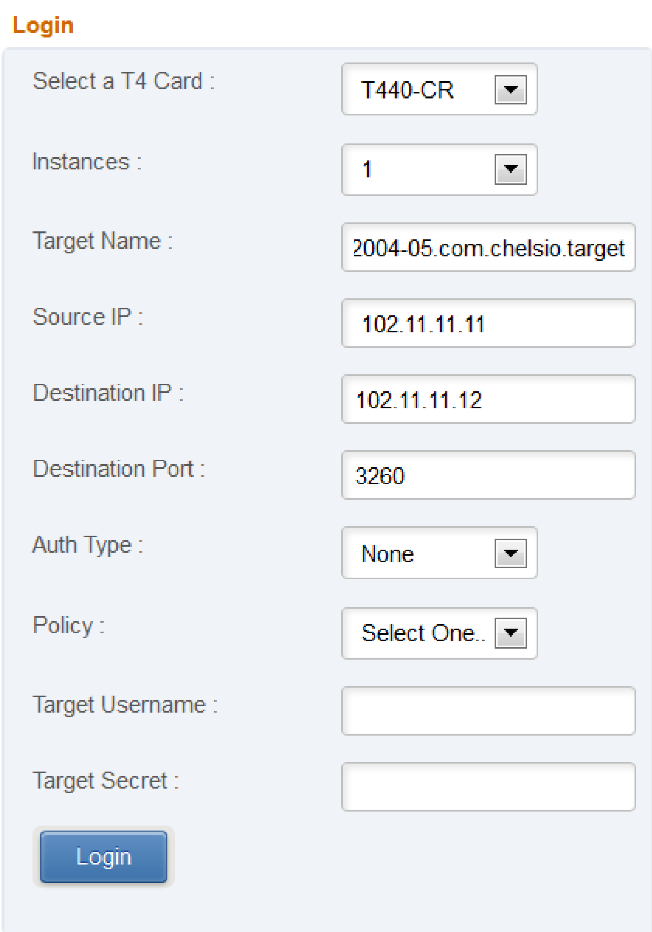


Figure 7.7.3 (f) - FO iSCSI Session Details module: Login

After successful login, details of the established iSCSI session will be displayed under the **Established sessions** section. Select the adapter and session id. Details of the selected session will be displayed. To end the session, click **Logout**.

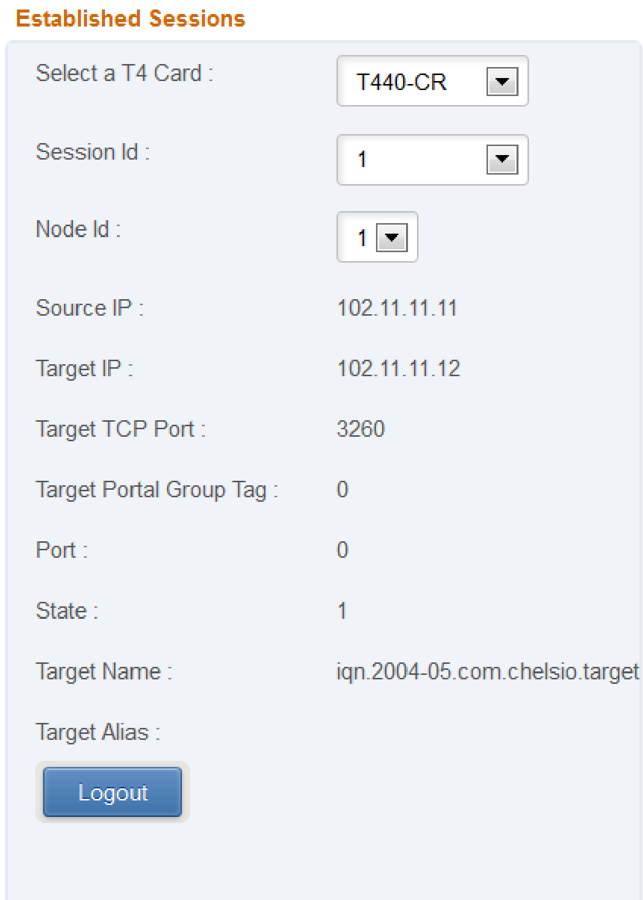


Figure 7.7.3 (g) - FO iSCSI Session Details module: Established Sessions

7.7.4. iSCSI Target (Linux)

This page allows to create new Targets and manage them (add/delete portals, add/delete LUNs, add/delete ACLs).It also provides information on Session details. Viewing and modifying Target properties is also available. The modules available on this page are as below:

Target Stack Globals

This module displays various global properties of a currently connected iSCSI target. Authentication priority between CHAP and ACL can be set here.

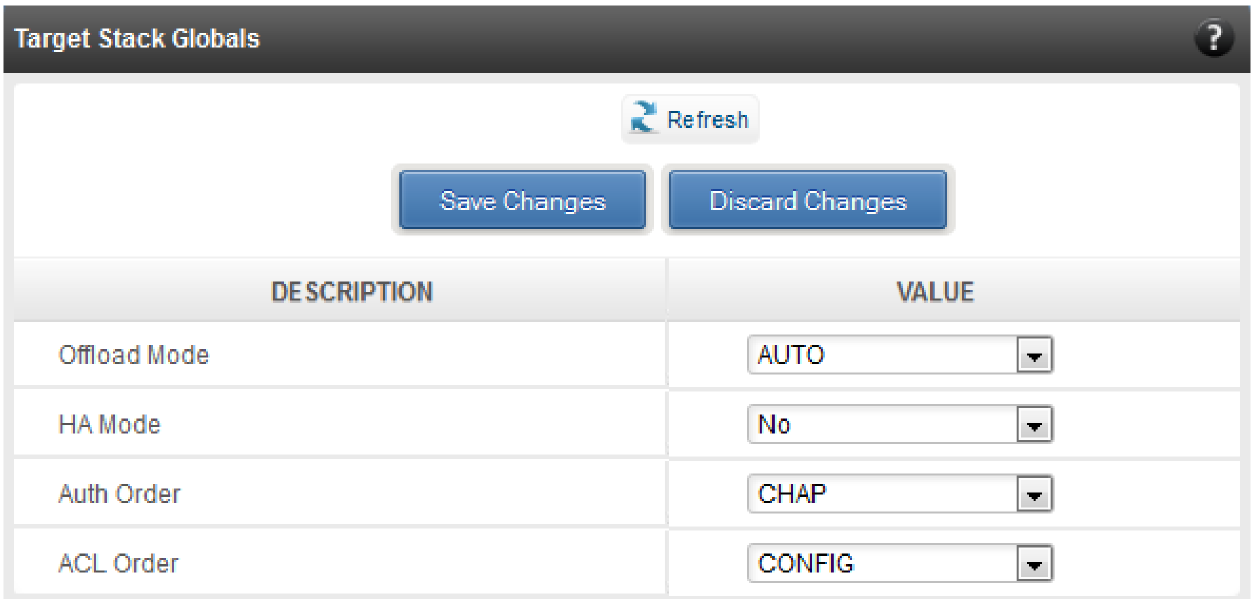


Figure 7.7.4 (a) - Target Stack Globals module

Target properties

Properties such as Target name and Alias, Max Data Receive Length, Authentication mode related to a specific iSCSI target can be viewed and modified here. iSCSI targets can be started/stopped or deleted.

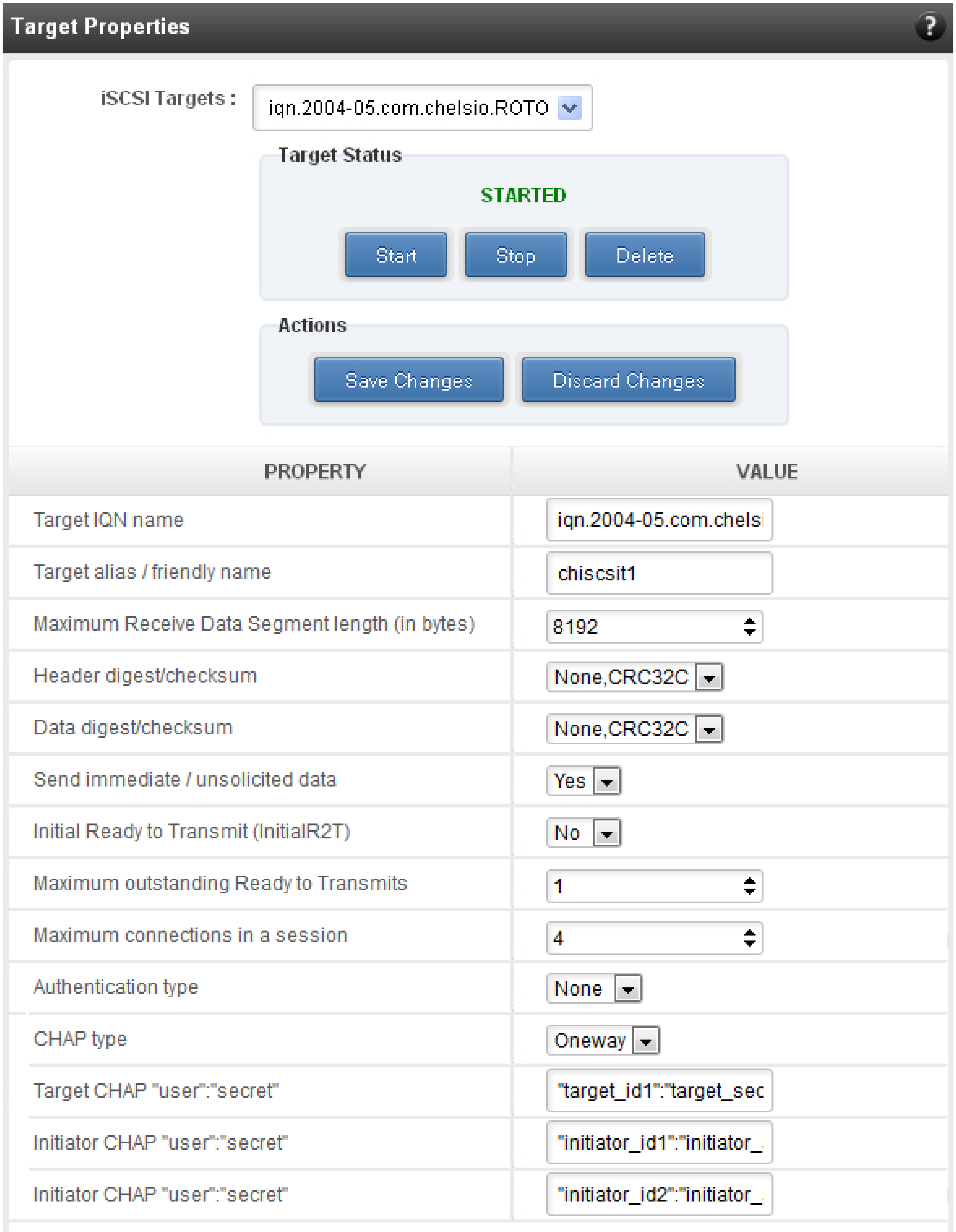


Figure 7.7.4 (b) - Target properties module

Session details

Details including Session ID, Initiator IQN and Connections List of all discovered and currently connected iSCSI targets are listed here.

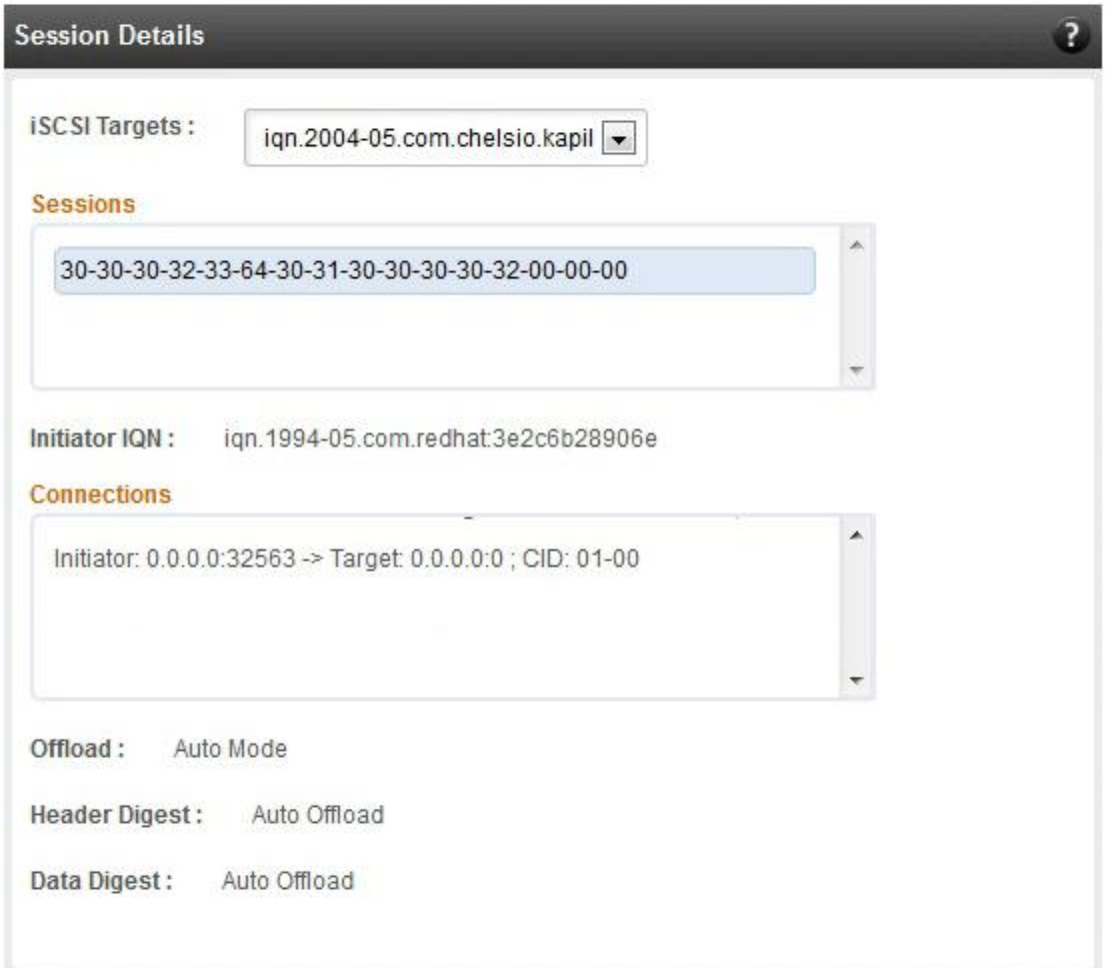


Figure 7.7.4 (c) - Session Details module

New Target Creation

New iSCSI target can be created here by specifying the Target IQN and Target Alias name.

Figure 7.7.4 (d) - New Target Creation module

7.7.5. LUNs

Various Logical Units created in an iSCSI Target can be managed here. The modules available on this page are as below:

- **View/Edit iSCSI Target LUNs**

This module displays various Logical Units created in an iSCSI Target. Selected LUNs can be deleted.

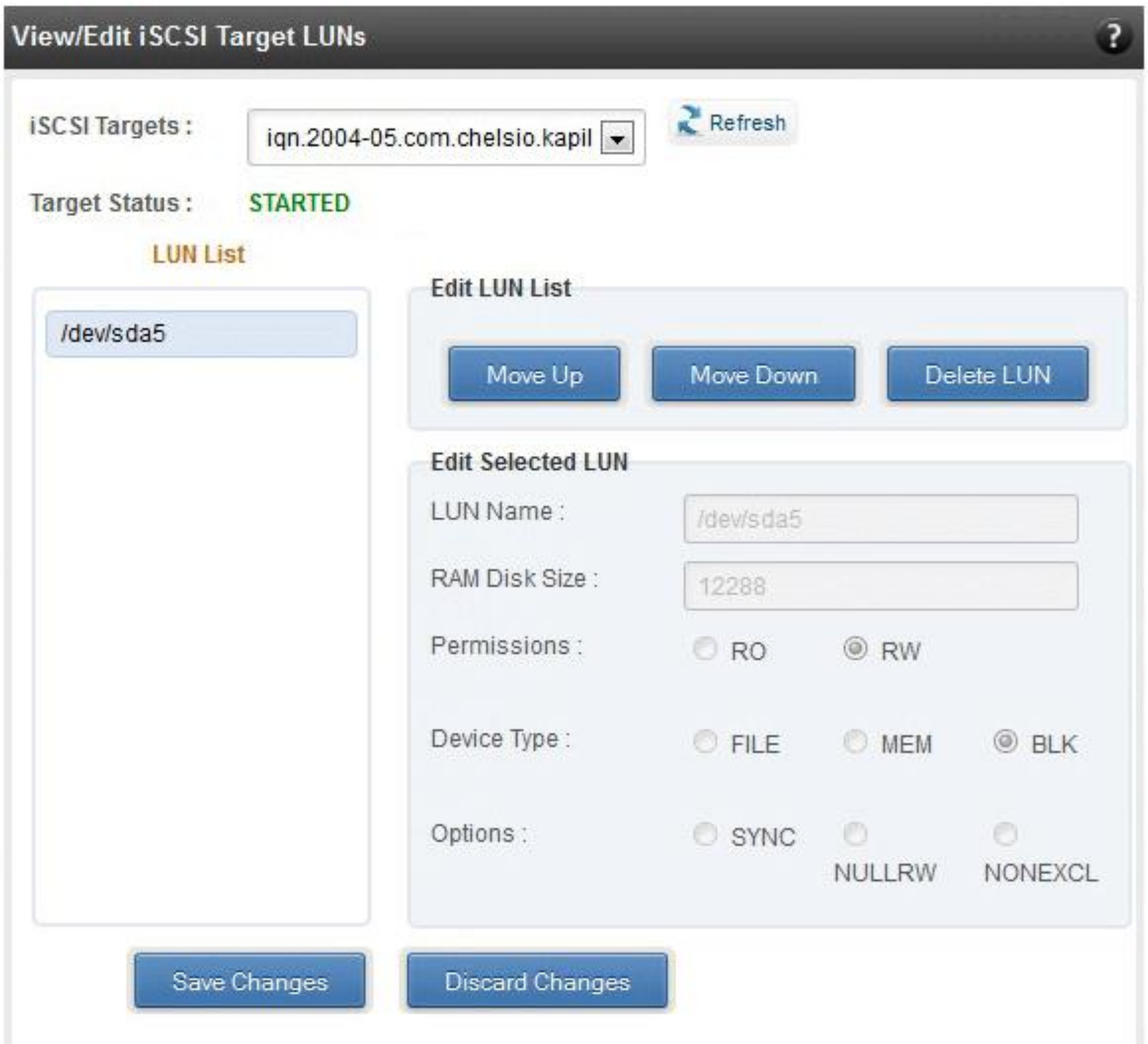


Figure 7.7.5 (a) - View/Edit iSCSI Target LUNs module

Add LUN

New LUNs can be added here by providing various parameters like Target Name, Target Device and RAM Disk Size etc. RW (Read-Write) and RO (Read Only) are the two kinds of permissions that can be set. If Ram Disk is selected, then a minimum of 16 MB should be provided.

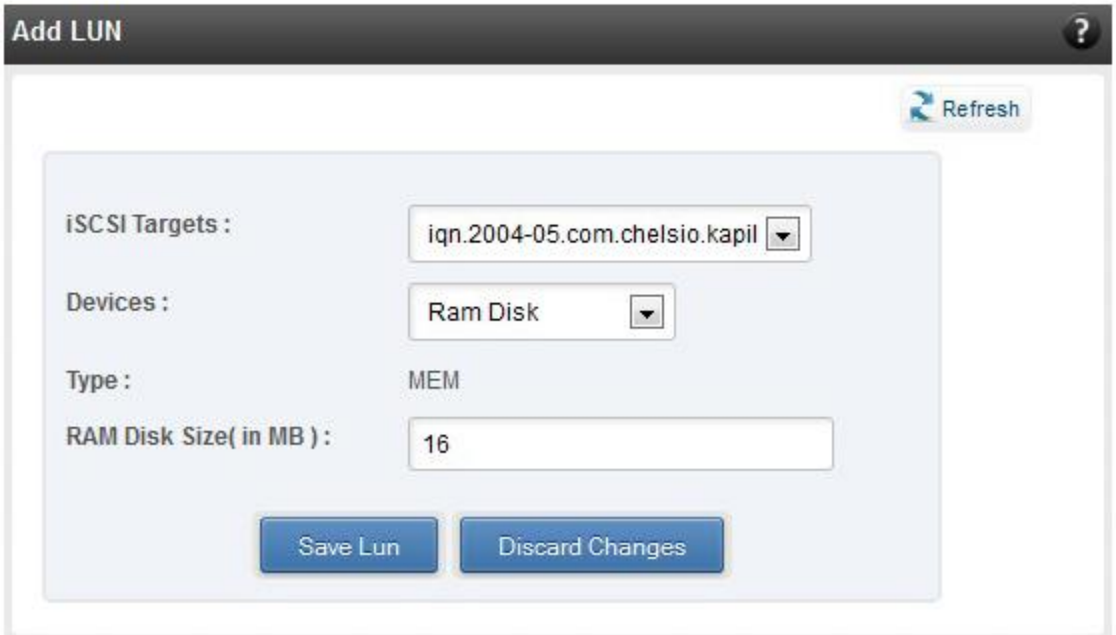


Figure 7.7.5 (b) - Adding a new LUN

7.7.6. Portal Groups

Portal details for currently connected iSCSI Targets can be viewed and added here. The modules available on this page are as below:

View/Edit iSCSI Target Portals

Portal List on the left displays details of the portal group on which an iSCSI target is listening and the related info is displayed on the right under Portal Details. Selected portals can be deleted.

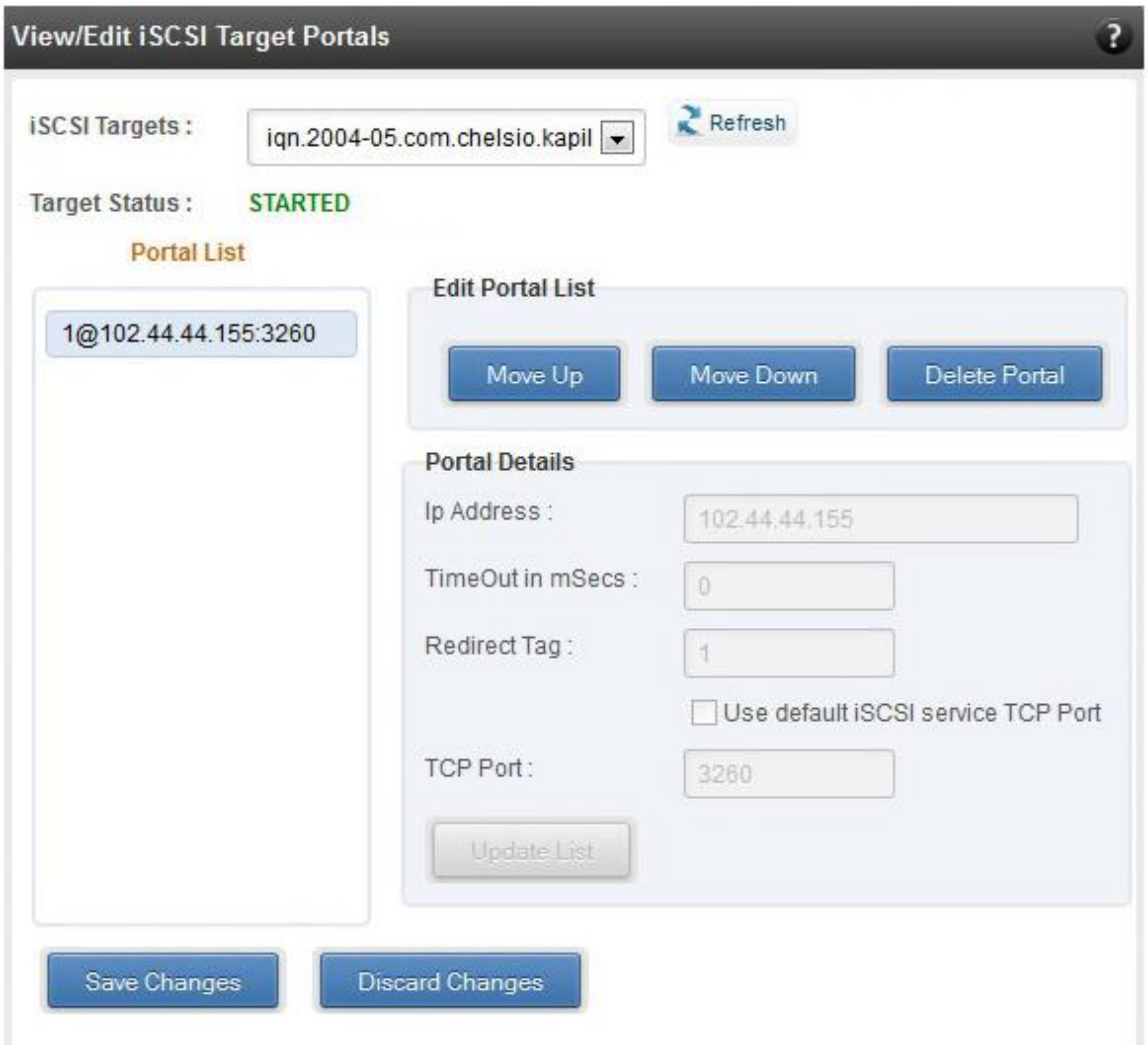


Figure 7.7.6 (a) - View/Edit iSCSI Target Portals module

Add Portal

New Portals can be added here by choosing the specific target and Portal IP address. The Port number should be 3260.

Figure 7.7.6 (b) - Adding a new Portal

7.7.7. ACLs

ACLs configured for currently connected iSCSI Targets can be managed here. The modules available on this page are as below:

View/Edit iSCSI Target ACLs

This module displays details for all the ACLs configured for an iSCSI Target. Selected ACLs can be deleted.

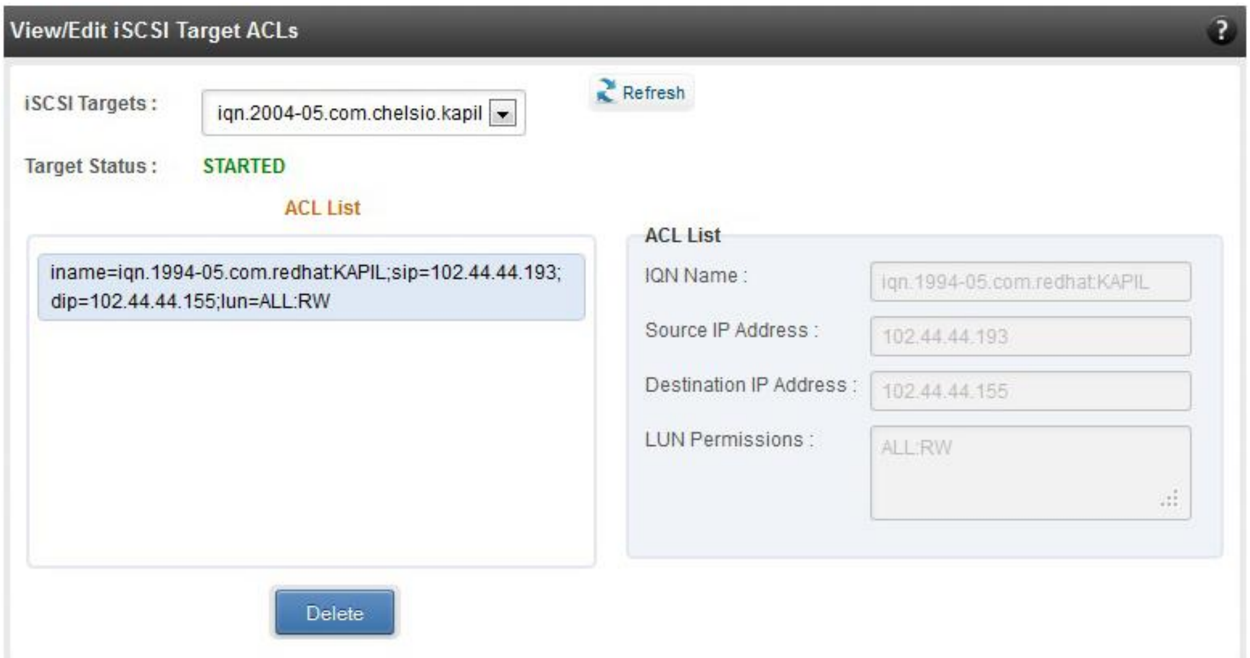


Figure 7.7.7 (a) - Target ACL operations module

Add ACL

New ACLs can be configured by specifying Target name, initiator IQN name, IP address and permission type.

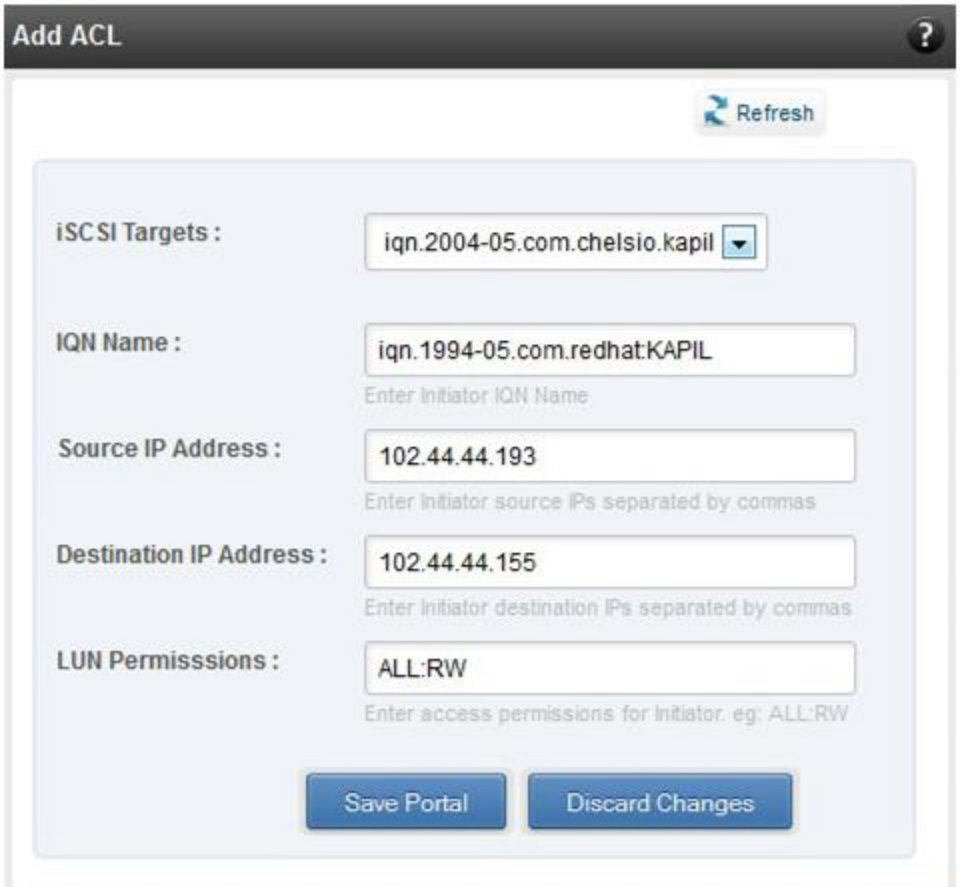


Figure 7.7.7 (b) - Adding new ACL

7.8. Hardware Features

The **Hardware** module lists the status of configuration modules under Hardware Features section, running on the agent.

Hardware

PROPERTY	VALUE
Boot service on server	Enabled
Filter service on server	Enabled
Traffic mgmt service on server	Enabled

Figure 7.8 – Hardware module for a Linux Agent

7.8.1. Filtering (Linux)

Filtering feature enhances network security by controlling incoming traffic as they pass through network interface based on source and destination addresses, protocol, source and receiving ports, or the value of some status bits in the packet. The modules available on this page are as below:

- **T3 Filtering configuration**

T3 Filtering options can be set only when offload driver (*t3_tom*) is not loaded.

This module lists various parameters which can be set while determining filtering options for a system IP. You can set the maximum number of filters and also add/delete filters. A filter with default values (the **Action** field set to **pass**; the **Protocol** field set to **any**) is created at the time of configuring the filtering module. To remove the default filter, enter 0 in the **Set Maximum Filters** field and click on **Set Filters**. The fields **IfName** and **FilterId** are mandatory. The **Action** field is set to **pass** and the **Protocol** field is set to **any** by default. Other possible values for the **Protocol** field are **tcp**, **udp** and **frag**. The **Priority** field can be used to determine the priority of a filter when Vlan ids are same. **Insert at position** features allows user to add a filter at a specified position.

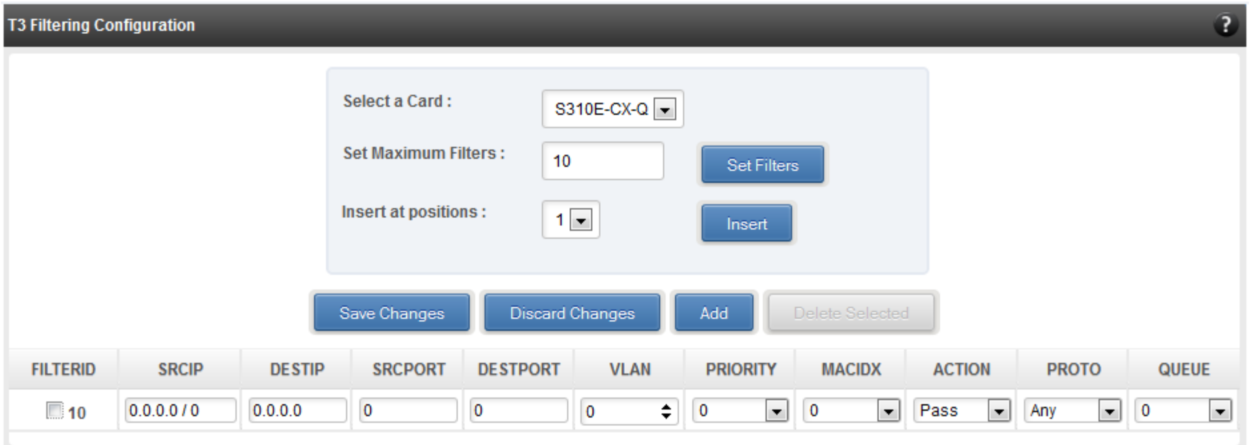


Figure 7.8.1(a) – T3 Filtering Configuration module

Note *Results for actions like adding a new filter or setting maximum filters make some time to reflect. Highlight the system item in the tree menu on the left, and click "Refresh system", to refresh data from the system, in case the updated settings are not being shown.*

T5/T4 Filtering configuration

Filtering options can be set only when offload driver (*t4_tom*) is not loaded.

A list of pre-defined filter selection combinations is displayed. The combination *fragmentation, mpshittype, protocol, vlan, port, fcoe* is active by default for T4 adapters. For T5 adapters, the default combination is *srvrsram, fragmentation, mpshittype, protocol, vlan, port, fcoe*. To select a different combination, highlight it in the **Combinations** list by clicking and click “Set Active Combination”.

You can create filter rules for any combination in the list. However, filter rule created only for the Active Combination will apply. To create a new rule, select the adapter type (T5 or T4) in the *Select a chip type* drop-down, select a combination and click “Add a Filter row”. The **FILTERID** and **T5/T4 CARD** fields are mandatory. After providing appropriate values for the parameters click “Save Changes”.

Note *For a detailed explanation regarding different fields, please refer **cxgbtool** manual by running* `man cxgbtool` *command on Management Agent CLI.*

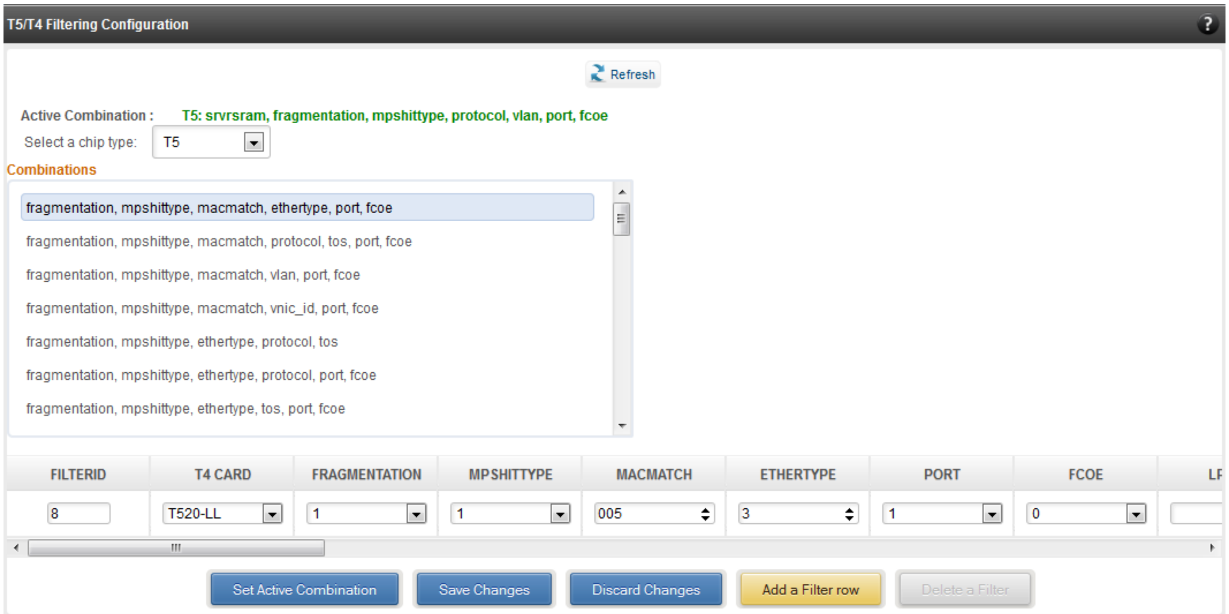


Figure 7.8.1(b) – T5/T4 Filtering Configuration module

7.8.2. Traffic Management (Linux)

Using this page, you can add/delete/modify offload policies only in the presence of offload driver (*t3_tom* for T3 adapters*; t4_tom* for T5 and T4 adapters).

Traffic Management configuration

The **Chelsio Card** section on the left displays all the cards available in the server and their corresponding policies on the right. Policies can be added and deleted. Policy Details displays the primitives (maximum 8) and actions which can be modified. For more details on creating policies, please refer to COP man pages.

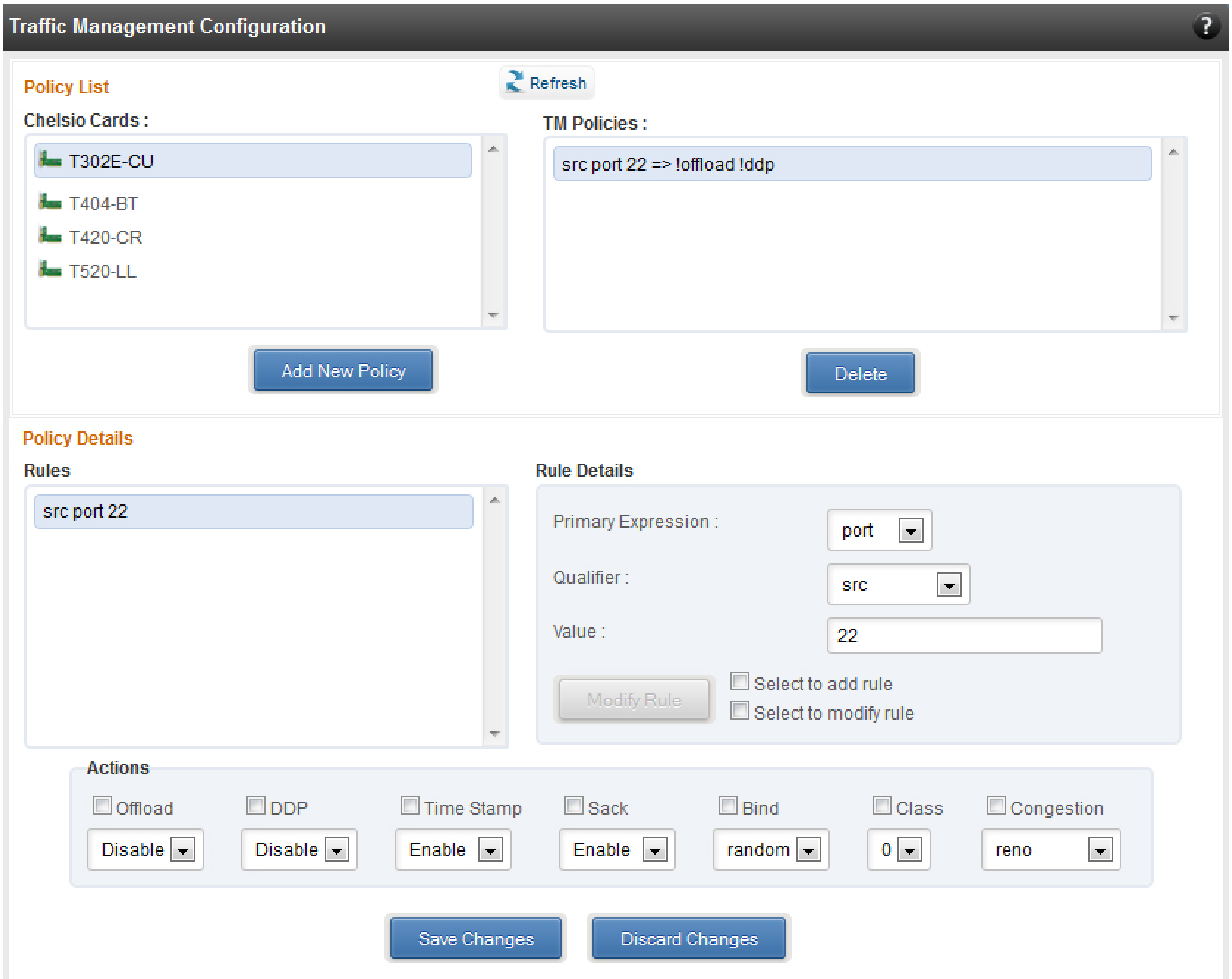


Figure 7.8.2 - Traffic Management Configuration module

7.8.3. Boot

- ### T4/T5 Save Config File (Linux)

This module displays the current T5/T4 configuration tuning option selected. You can also change the tuning option by selecting the config file for each option located in /<driver_package>*/src/network/firmware*. For instance, to select *Low latency Networking* for T4 adapter, locate the file, *t4-config.txt, in* /<driver_package>*/src/network/firmware/low_latency_config* directory.

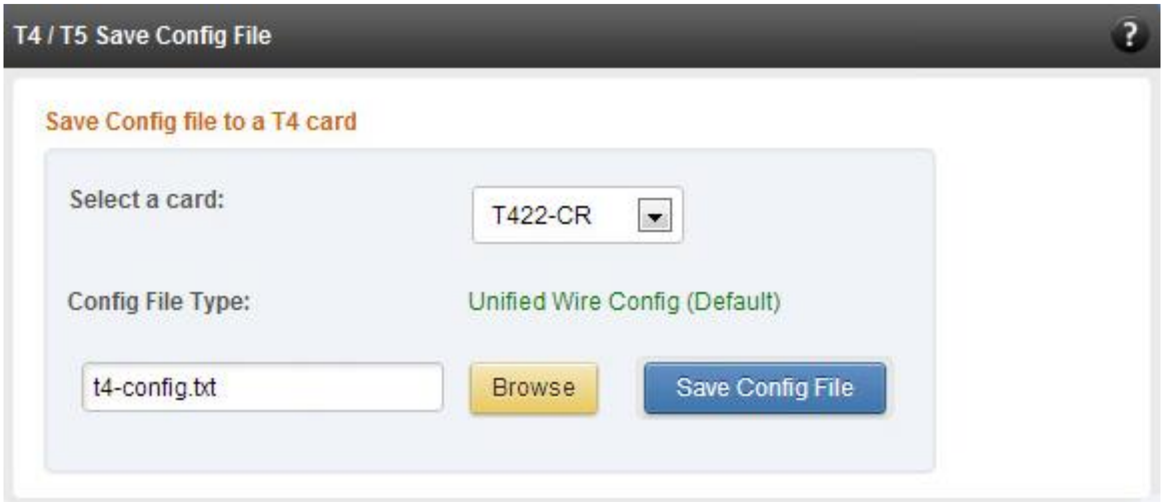


Figure 7.8.3 (a) – T4/T5 Save Config File module

T5/T4 Boot Option ROM management

This module allows managing the PXE and FCoE boot capability for Chelsio T5 and T4 adapters. The Option ROM (PXE and FCoE) may be installed to or erased from the card. The version of Option ROM flashed can be viewed here.

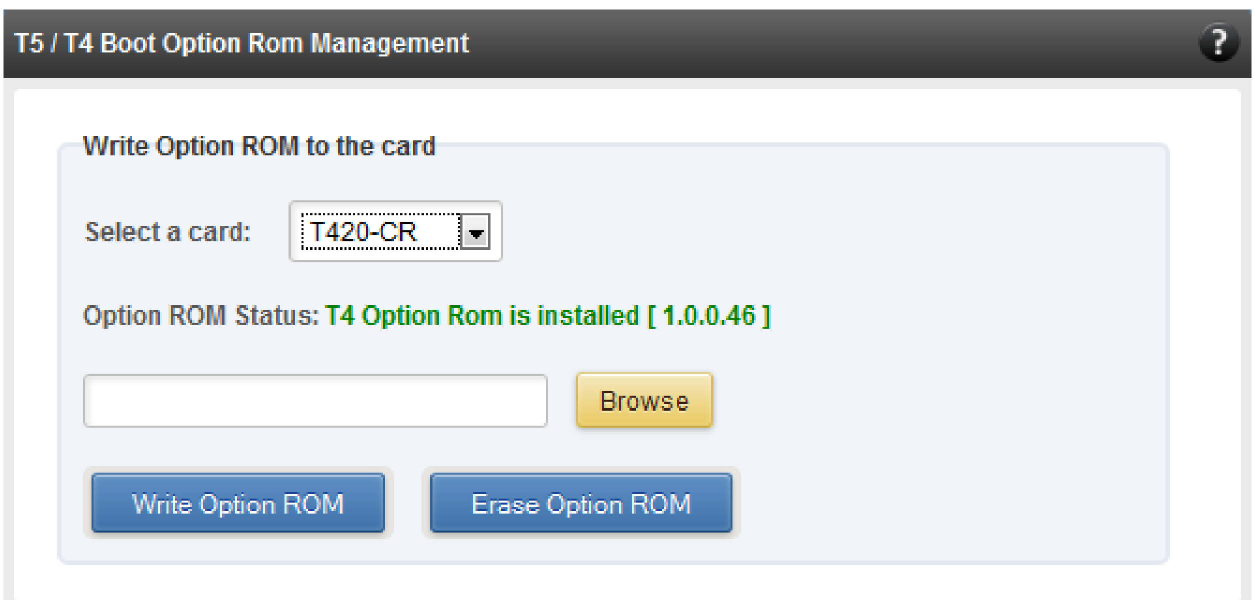


Figure 7.8.3 (b) – T4/T5 Option ROM Management module

T5/T4 Boot Configuration

This module can be used to view and configure PXE, FCoE and iSCSI Option ROM settings for Chelsio T5 and T4 adapters.

PXE physical functions and order of ports for PXE boot can be selected using the **PXE** option. You can also enable/disable PXE BIOS and set VLAN.

The **FCoE** option can be used to configure FCoE Option ROM settings. Using the *Function* parameter, you can set port order for target discovery and discovery timeout. The *Boot* parameter can be used to discover targets and view properties of LUNs assigned to the targets. Clicking on the **Discover Targets** button will list all the discovered targets and clicking on target will list assigned LUNs on the right. Select a LUN from the list to view details. *Show WWPN* parameter will display the WWPNs of all the ports.

There are four configurable parameters available under the **iSCSI** option: *Function*, *Initiator*, *Network* and *Boot Devices*. Using the *Function* parameter you can enable/disable BIOS, set port order for target discovery and discovery timeout. The *Initiator* parameter allows you to configer initiator properties like IQN name, header digest and data digest. You can also set CHAP authentication method or disable it. The *Network* parameter allows you to configure various settings on the port like enabling/disabling IPv6 support, specifying initiator IP type, etc. Using the *Boot Devices* option you can set various iSCSI target properties.

Note *Enable Option ROM only if you are planning to boot the system via PXE or install the operating system on discovered iSCSI or FCoE LUN.*

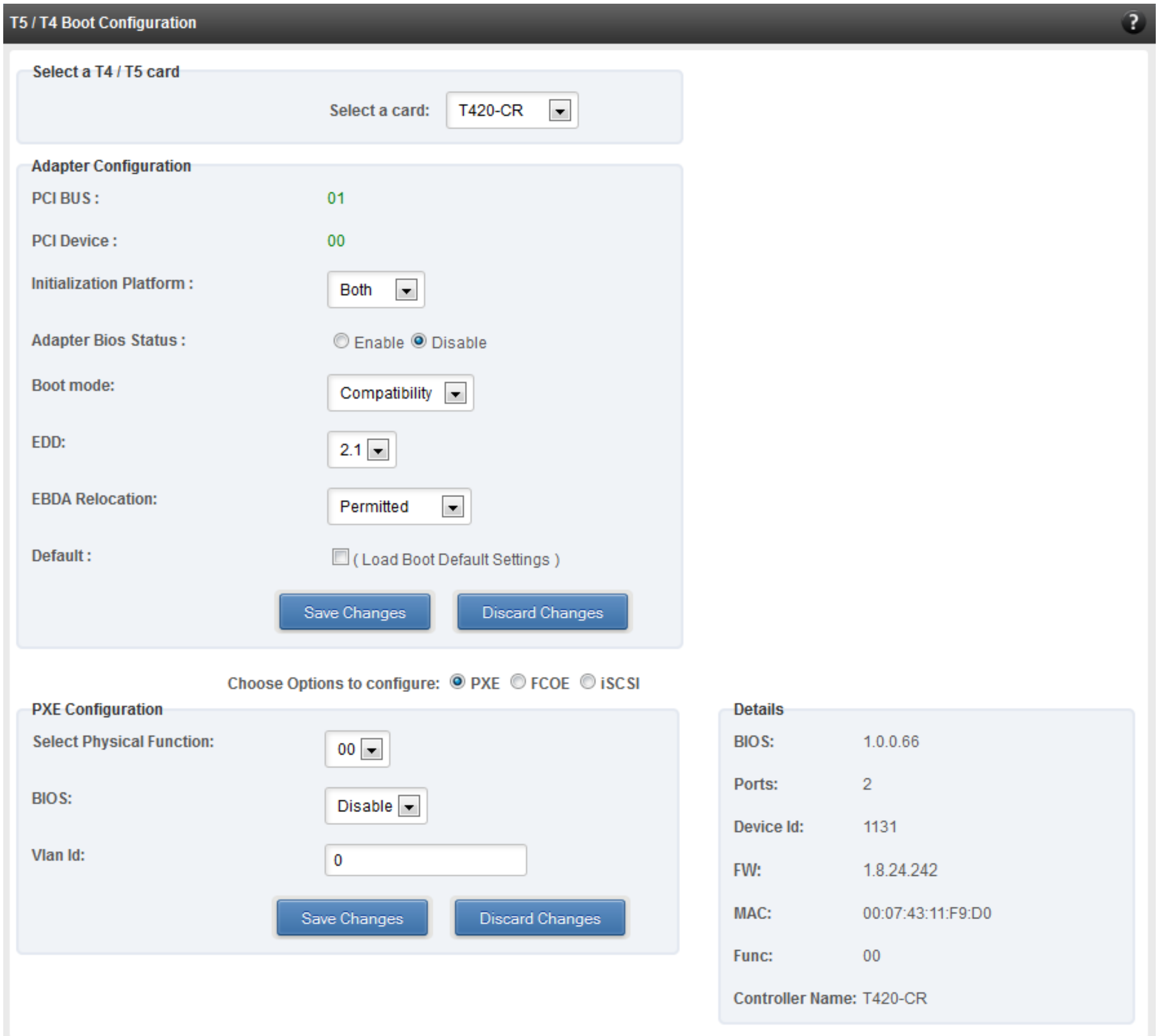


Figure 7.8.3 (c) - PXE Boot configuration for T4 CNAs

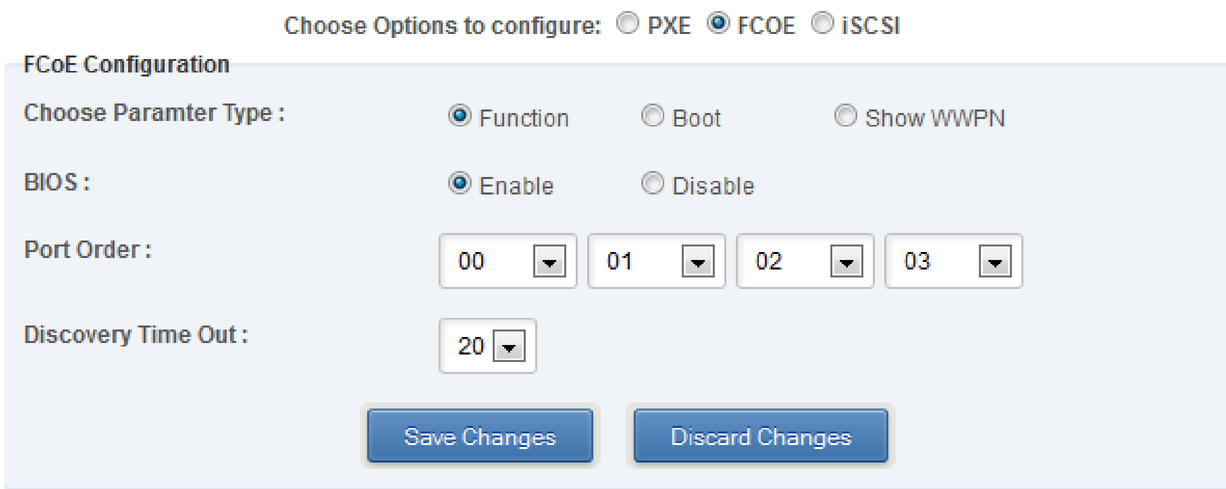


Figure 7.8.3 (d) - FCoE Boot configuration for T4 CNAs: Function parameter

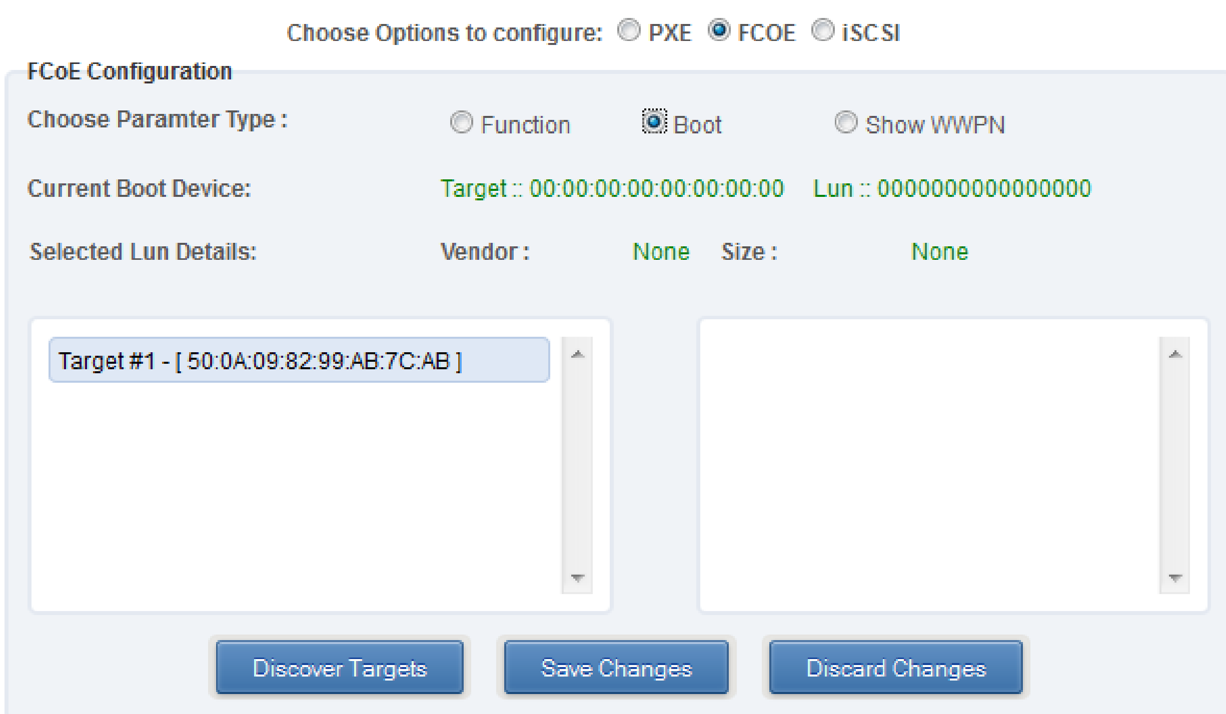


Figure 7.8.3 (e) - FCoE Boot configuration for T4 CNAs: Boot parameter

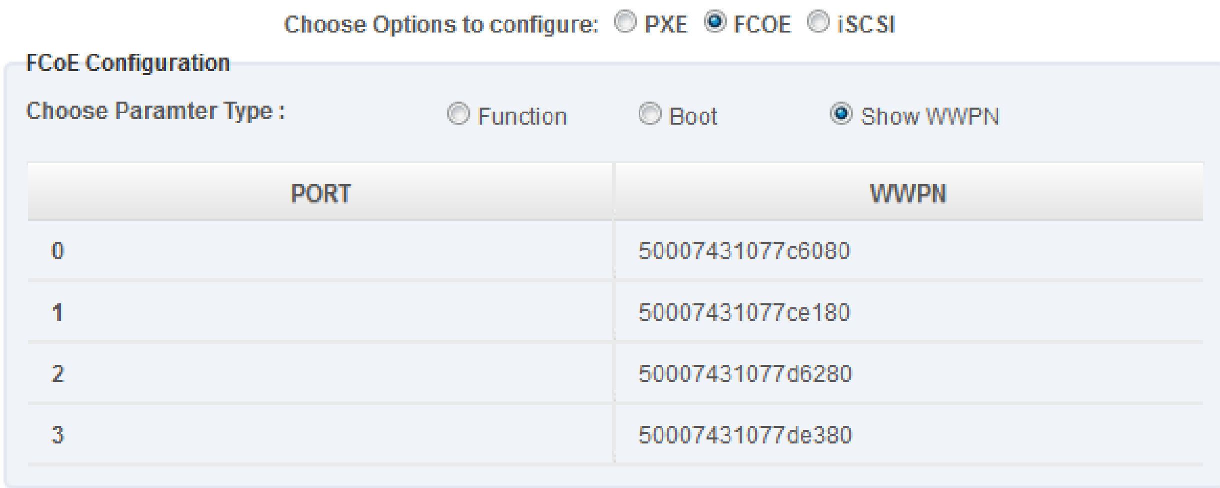


Figure 7.8.3 (f) - FCoE Boot configuration for T4 CNAs: Show WWPN parameter

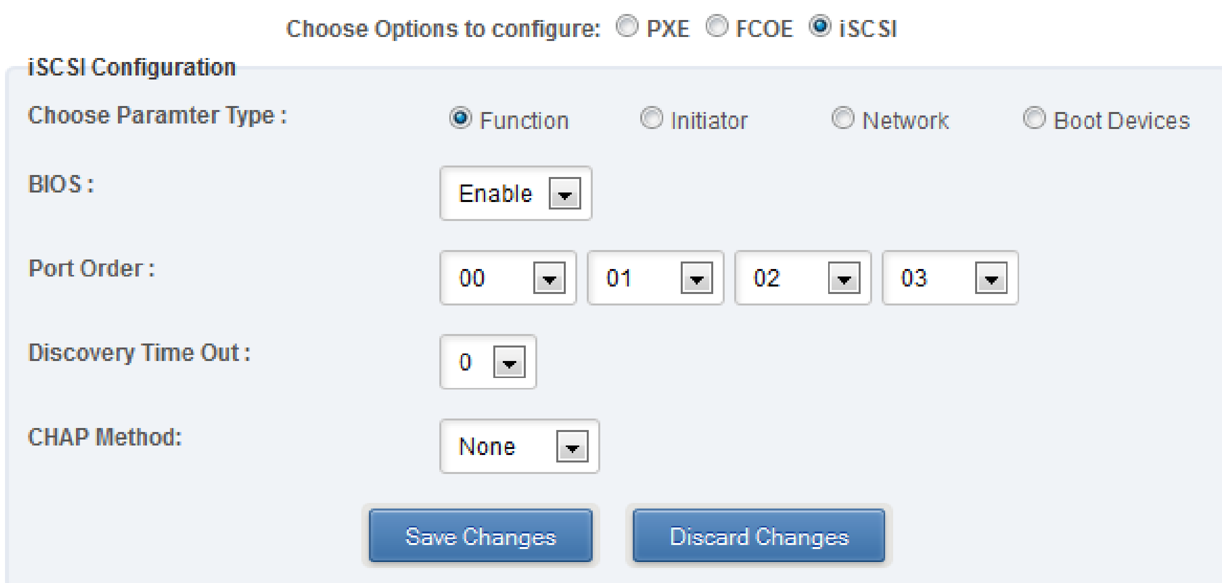


Figure 7.8.3 (g) - iSCSI Boot configuration for T4 CNAs: Function parameter

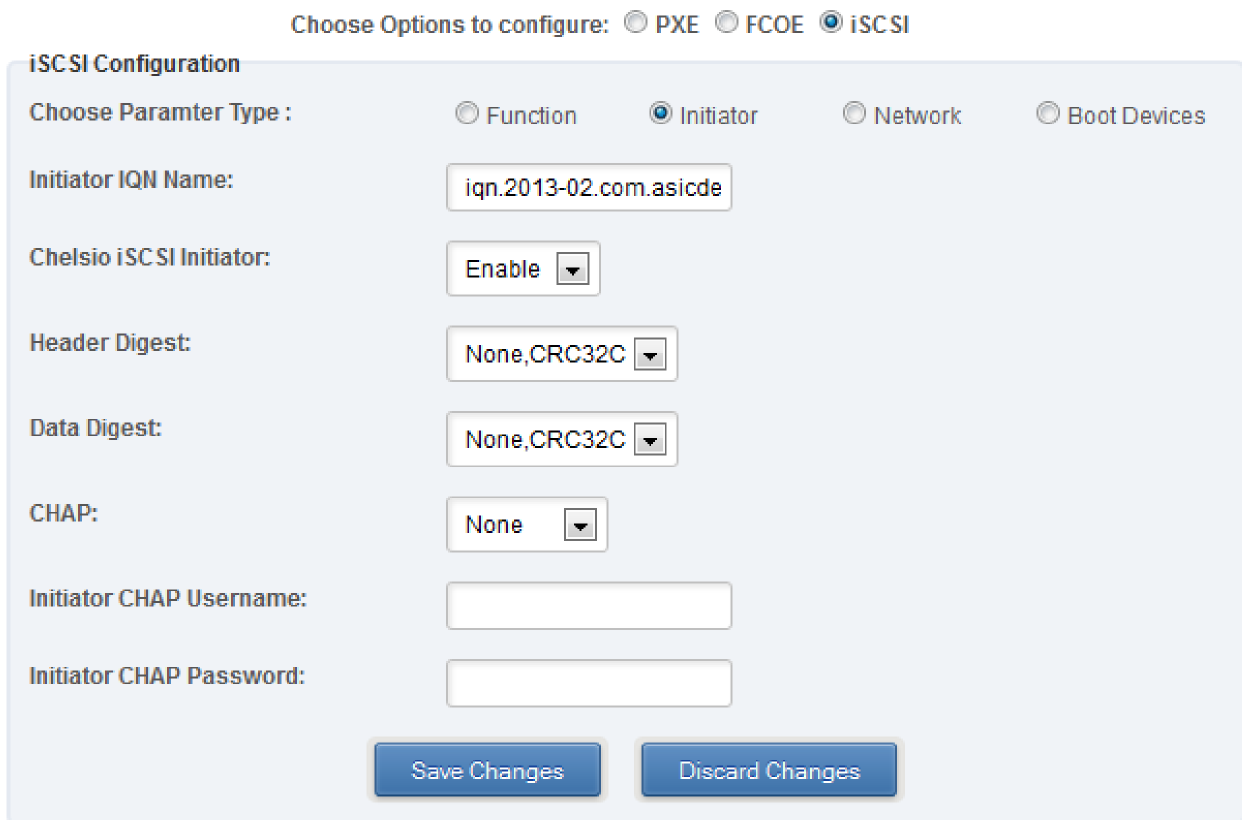


Figure 7.8.3 (h) - iSCSI Boot configuration for T4 CNAs: Initiator parameter

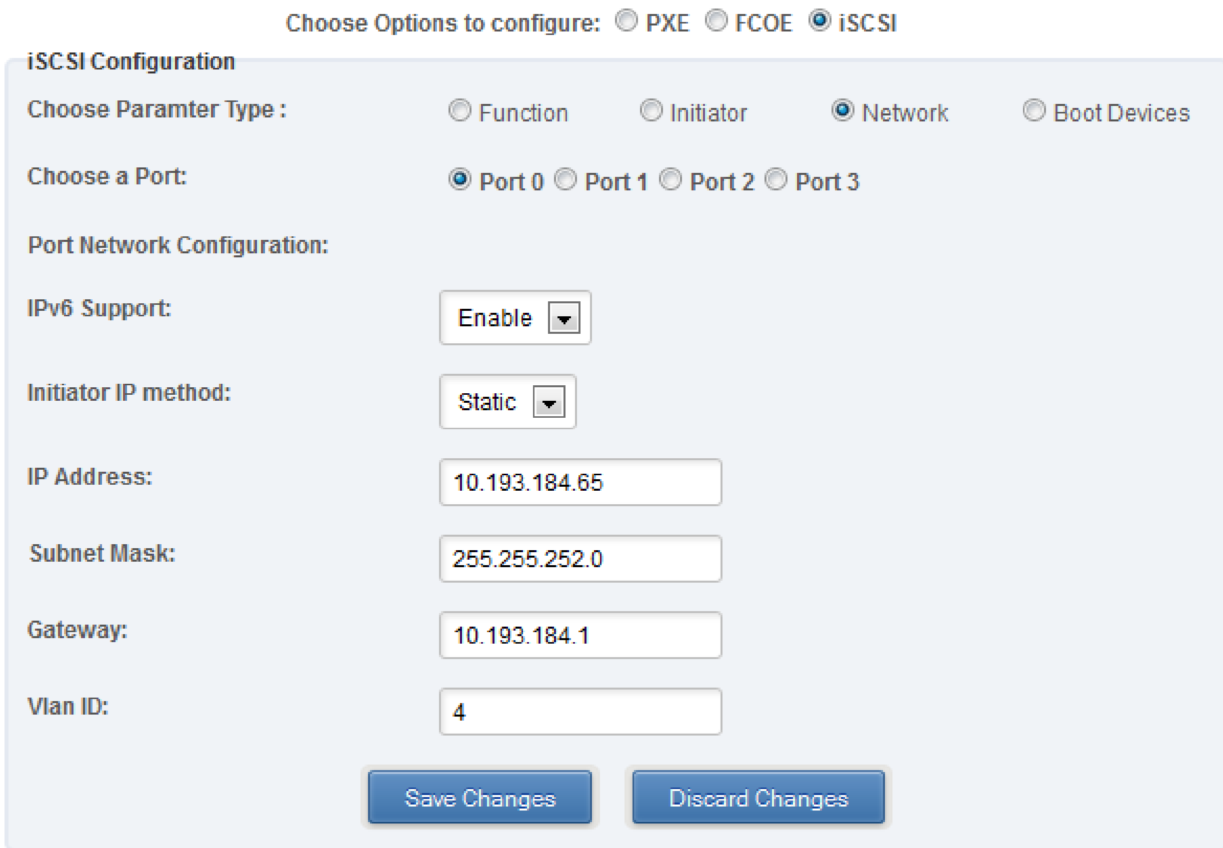


Figure 7.8.3 (i) - iSCSI Boot configuration for T4 CNAs: Network parameter

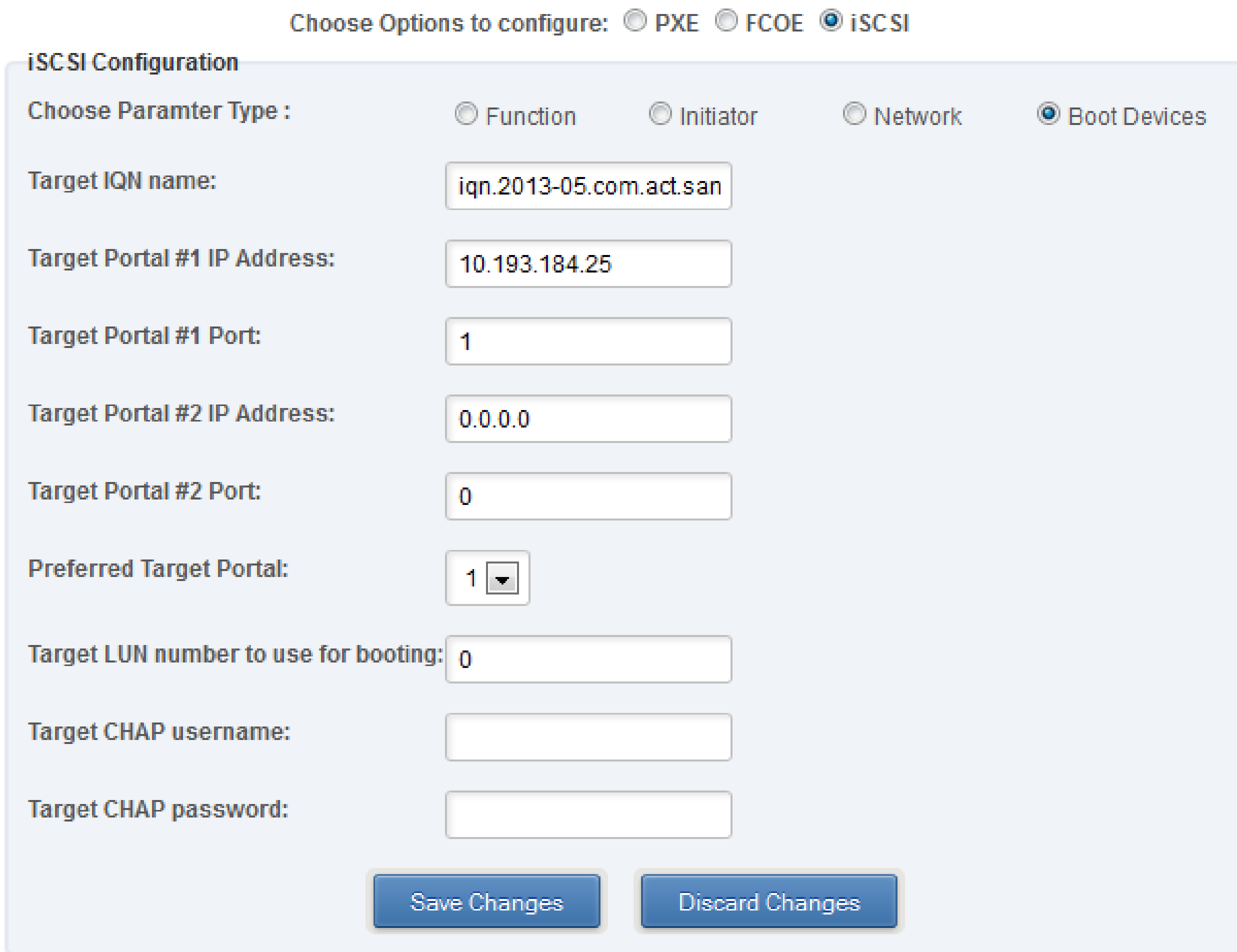


Figure 7.8.3 (j) - iSCSI Boot configuration for T4 CNAs: Boot Devices parameter

7.8.4. Bypass

You can use the **Bypass** page to configure various settings for Chelsio's bypass adapters like setting bypass operation mode, creating rules (filters), starting/stopping BA server, etc. There are two modules available: **Bypass Configuration** and **Redirect Configuration**.

Bypass Configuration

In the **Bypass Configuration** module, you can view the status and start/stop the BA server accordingly. The adapter will redirect packets using the mode speciffied in the **Default bypass mode** field unless otherwise specified in the **Current bypass mode**.

The **Watchdog** timer is used to ensure that if there is a software failure, the switch will enter the default state. The **Watchdog timeout** value should be provided in milliseconds.

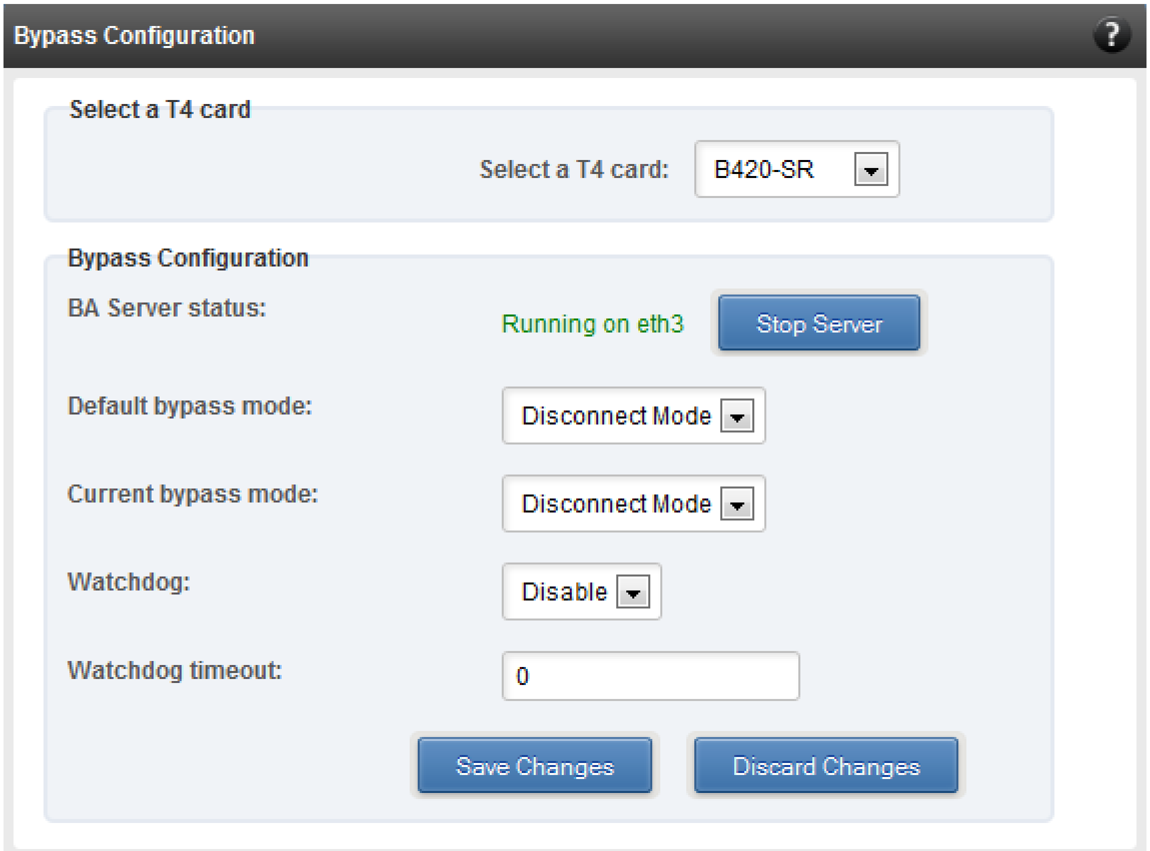


Figure 7.8.4 (a) - Bypass Configuration module

Redirect Configuration

In the **Redirect Configuration** module, you can set **rules** (filters), based on which the bypass adapter will redirect packets. You can group **rules** into **tables**. You can save the currently configured tables and rules for a bypass adapter into a shell script using the **Download Configuration** button.

The **Table Configuration** tab displays BA server status and the number of tables created. You can create new tables or perform various actions on the existing ones.

- **Delete table**: Delete the selected table and all the rules present in it.
- **Purge table**: Delete all the rules present in the selected table. This action will not delete the table.
- **Activate table**: Enable the selected table.
- **Deactivate table**: Disable the selected table.
- **Create table**: Create a new table. The new table created will be inactive by default. Use the **Activate table** option to enable it. You can cretate upto 5 tables.

In the **Rules Configuration** tab, you can add, delete and configure rules. Use the **Add a Filter row** button to add a new rule by specifying the rule id in the *INDEX* field and providing the required parameters. Finally, click **Save** Changes.

To edit an existing rule, select the corresponding checkbox, change the desired paremeters and click **Save Changes**.

To delete a rule, select the corresponding checkbox and click **Delete a Filter.** Finally, click **Save Changes**. You can delete multiple rules using this method.

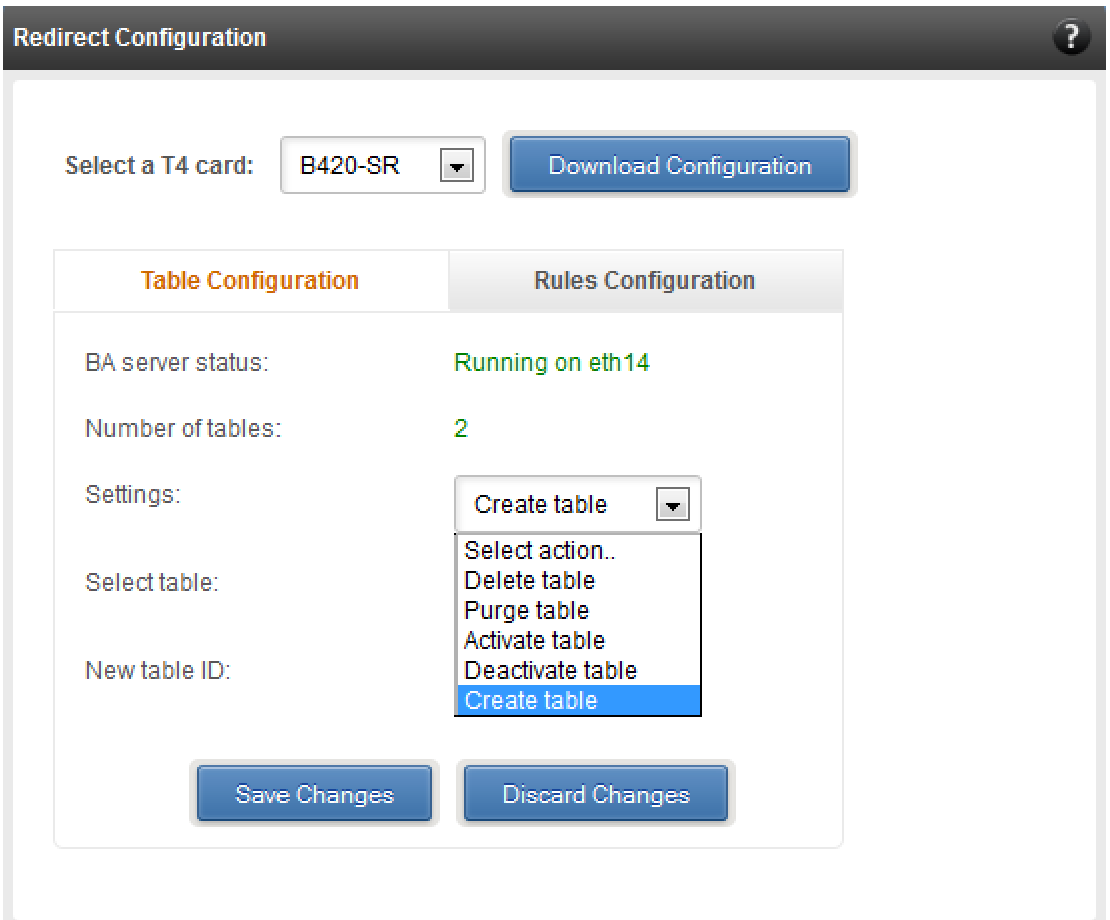


Figure 7.8.4 (b) - Redirect Configuration module: Table configuration tab

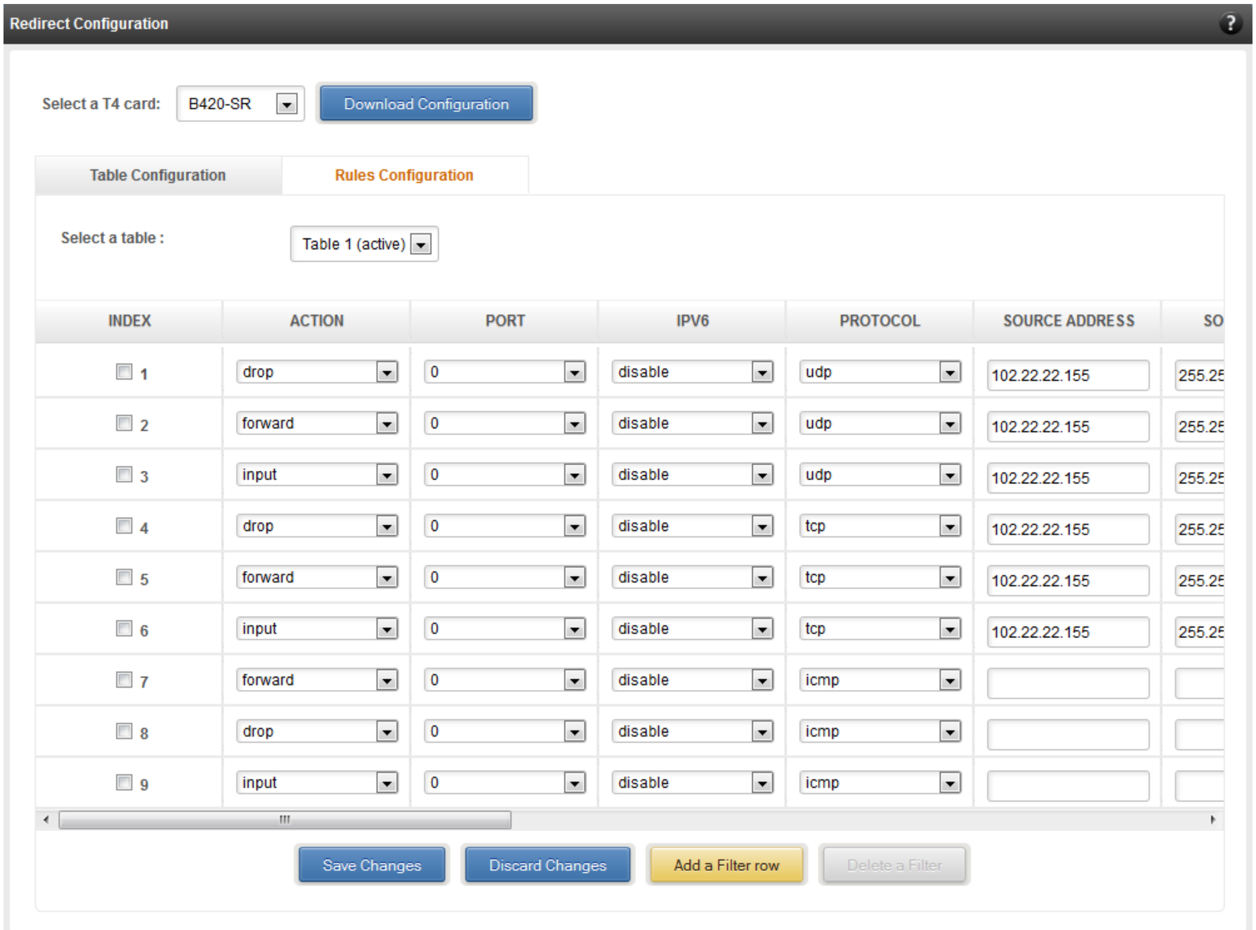


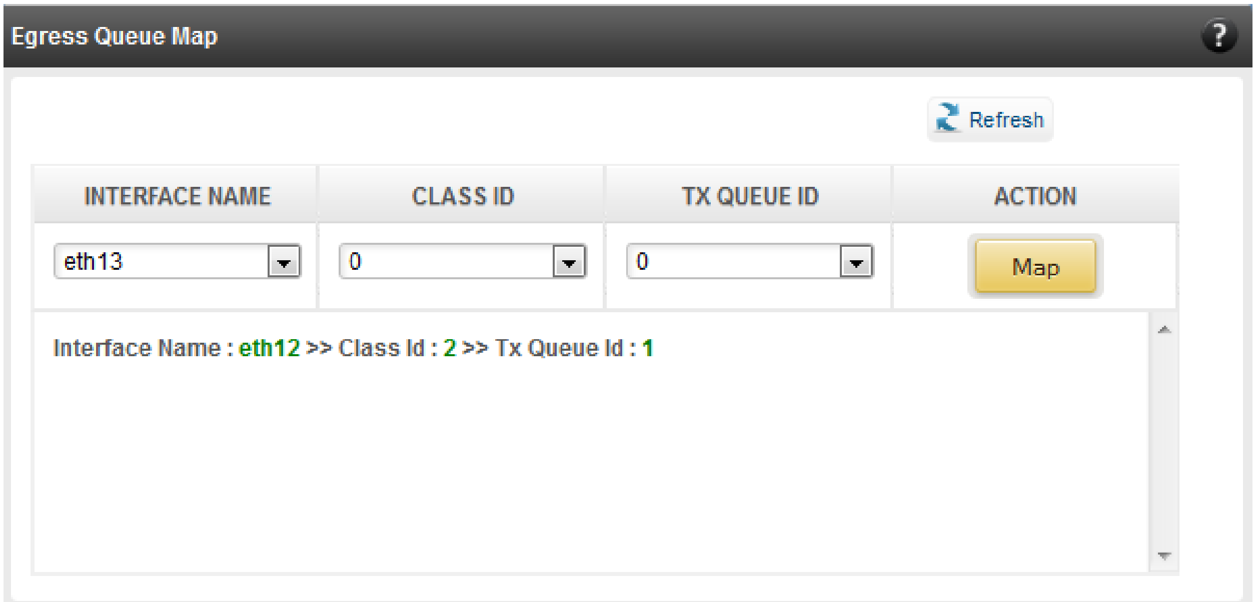
Figure 7.8.4 (c) - Redirect Configuration module: Rules configuration tab

7.8.5. T4 Egress Class Schedulers

Schedulers can be set only when T5/T4 network driver (*cxgb4*) is loaded.

- **Egress Queue Map**

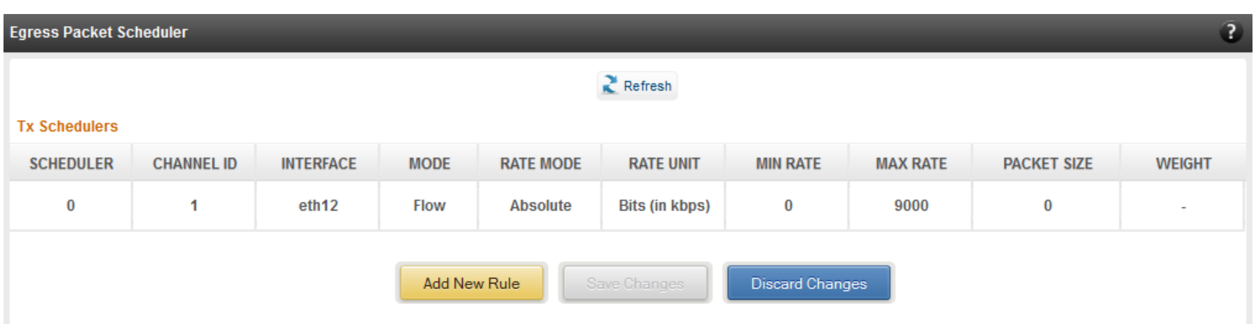
Using this module, you can bind (map) NIC (non-offloaded) Tx queues to Tx Scheduler classes.



7.8.5 (a) – Egress Queue Map module

- **Egress Packet Scheduler**

Using this module you can configure different scheduler hierarchy levels (i.e.Class Rate Limiting, Class Weighted Round Robin and Channel Rate Limiting). Based on the parameters specified, different scheduler levels can be configured. To know more about the levels and related parameters, please refer **Traffic Management** chapter (Click here).



7.8.5 (b) – Egress Packet Scheduler module

8. Uninstalling Unified Wire Manager

This section describes the method to uninstall components of Chelsio Unified Manger.

8.1. Uninstalling Management Agent

- Use the following query command to determine the name of the agent RPM:

```
[root@host~]# rpm -qa | grep chelsio-uwire_mgmt-agent
```

- Now, execute the following command with the result from the above query to uninstall Management Agent:

E.g. for RHEL 6.3:

```
[root@host~]# rpm -e chelsio-uwire_mgmt-agent-rhel6u3-2.2-xyz.x86_64
```

8.2. Uninstalling Management Client

1. Use the following query command to determine the name of the client RPM:

```
[root@host~]# rpm -qa | grep chelsio-uwire_mgmt-client
```

2. Now, execute the following command with the result from the above query to uninstall Management Client:

E.g. for RHEL 6.3:

```
[root@host~]# rpm -e chelsio-uwire_mgmt-client-rhel6u3-2.2-xyz.x86_64
```

8.3. Uninstalling Management Station

1. Use the following query command to determine the name of the Management Station RPM:

```
[root@host~]# rpm –qa | grep chelsio-uwire_mgmt-station
```

2. Now, execute the following command with the result from the above query to uninstall Management Station:

E.g. for RHEL 6.3:

```
[root@host~]# rpm –e chelsio-uwire_mgmt-station-rhel6u3-2.2-xyz.x86_64
```

XXII. Unified Boot

1. Introduction

PXE is short for Preboot eXecution Environment and is used for booting computers over an Ethernet network using a Network Interface Card (NIC). FCoE SAN boot process involves installation of an operating system to an FC/FCoE disk and then booting from it. iSCSI SAN boot process involves installation of an operating system to an iSCSI disk and then booting from it.

This section of the guide explains how to configure and use Chelsio Unified Boot Option ROM which flashes PXE, iSCSI and FCoE Option ROM onto Chelsio's converged network adapters (CNAs). It adds functionalities like PXE, FCoE and iSCSI SAN boot.

This section of the guide also describes the use and configuration of Chelsio's DUD for OS installations via PXE server on FC/FCoE LUN and iSCSI LUN. This solution can be used for installing operating systems over an Ethernet network/SAN using Chelsio's T5 and T4 based Converged Network adapters (CNAs).

1.1. Hardware Requirements

1.1.1. Supported platforms

Following is the list of hardware platforms supported by Chelsio Unified Boot software:

- DELL PowerEdge T710
- DELL PowerEdge 2950
- DELL PowerEdge T110
- Dell T5600
- IBM X3650 M2
- IBM X3650 M4*
- HP ProLiant DL385G2
- Supermicro X7DWE
- Supermicro X8DTE-F
- Supermicro X8STE
- Supermicro X8DT6
- Supermicro X9SRL-F
- Supermicro X9SRE-3F
- ASUS P5KPL
- ASUS P8Z68

* If system BIOS version is lower than 1.5 and both Legacy and uEFI are enabled, system will hang during POST. Please upgrade the BIOS version to 1.5 or higher to avoid this issue.

1.1.2. Supported Switches

Following is the list of network switches supported by Chelsio Unified Boot software:

- Cisco Nexus 5010 with 5.1(3) N1 (1a) firmware.
- Arista DCS-7124S-F
- Mellanox SX_PPC_M460EX

Other platforms/switches have not been tested and are not guaranteed to work.

1.1.3. Supported adapters

Following are the currently shipping Chelsio adapters that are compatible with Chelsio Unified Boot software:

- T62100-LP-CR
- T6225-CR
- T580-OCP-SO*
- T520-OCP-SO*
- T520-BT
- T580-CR
- T520-LL-CR
- T520-SO-CR*
- T520-CR
- T540-CR
- T580-LP-CR
- T580-SO-CR*

* Only PXE supported

1.2. Software Requirements

Chelsio Unified Boot Option ROM software requires Disk Operating System to flash PXE ROM onto Chelsio adapters.

The Chelsio Driver Update Disk driver has been developed to run on 64-bit Linux platforms. Following is the list of Drivers/Software and supported Linux distributions:

Linux Distribution	Driver/Software (DUDs)
RHEL 7.3, 3.10.0-514.el7	PXE, FCoE, iSCSI
RHEL 7.2, 3.10.0-327.el7	
RHEL 6.8, 2.6.32-642.el6	
RHEL 6.7, 2.6.32-573.el6	

SLES 12 SP2, 4.4.21-69-default	
SLES 12 SP1, 3.12.49-11-default	
SLES 11 SP4, 3.0.101-63-default	

2. Flashing firmware and option ROM

Depending on the boot mode selected, Chelsio Unified Boot provides two methods to flash firmware and option ROM onto Chelsio adapters: Flash utility *cfut4* for Legacy mode and *HII* for uEFI mode. Both methods also provide the functionality to update/erase Boot configuration, Hardware configuration and Phy Firmware files.

2.1. Preparing USB flash drive

This document assumes that you are using a USB flash drive as a storage media for the necessary files. Follow the steps below to prepare the drive:

i. Create a DOS bootable USB flash drive. (Click here for instructions)

ii. Create a directory *CHELSIO* on the USB flash drive.

iii. If you haven't done already, download the driver package from Chelsio Download Center, service.chelsio.com

iv. Untar the downloaded package and change your working directory to *OptionROM* directory.

```
[root@host~]# cd <driver_package>/Uboot/OptionROM
```

v. Copy all the files and place them in the *CHELSIO* directory created on the USB flash drive.

vi. Plug-in the USB flash drive in the system on which the Chelsio CNA is installed.

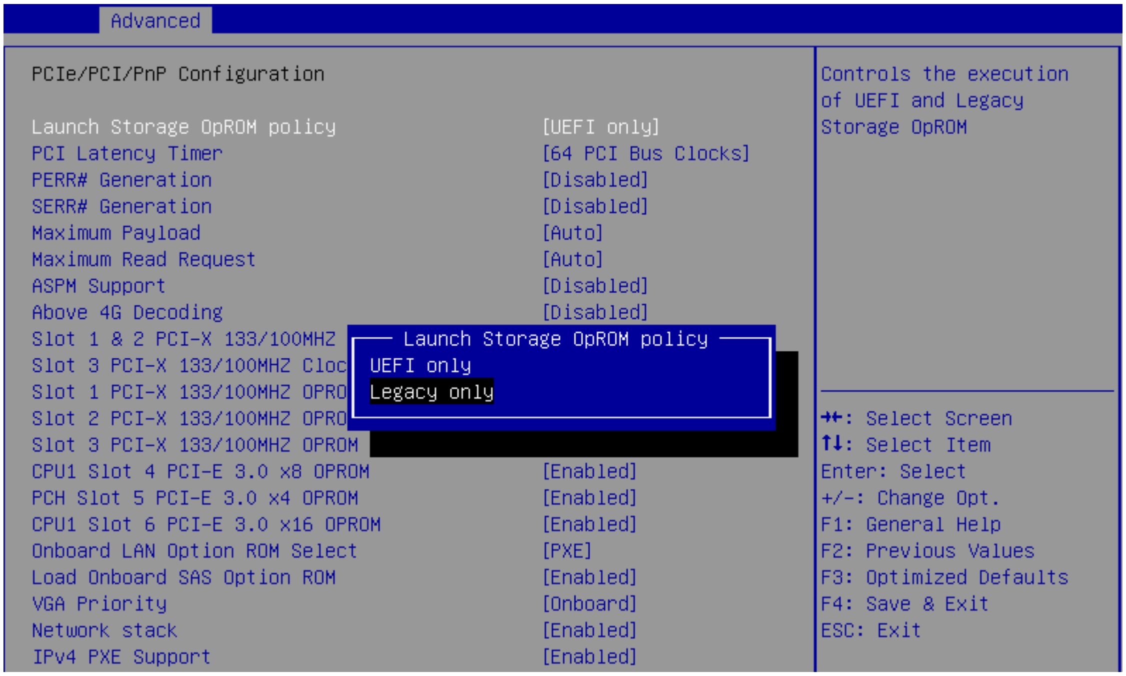
vii. Reboot the system and go into the BIOS setup.

viii. Make the USB flash drive as the primary boot device.

ix. Save the changes.

2.2. Legacy

i. In BIOS, configure the system having Chelsio CNA to boot in Legacy mode.

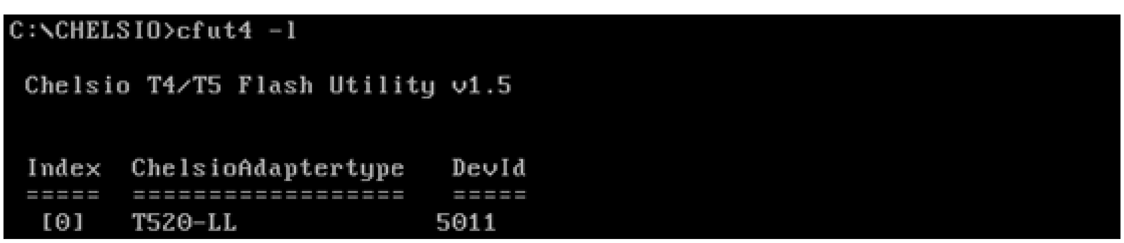


ii. Once the system boots from the USB flash drive, change your working directory to *CHELSIO* directory:

```
C:\>cd CHELSIO
```

iii. Run the following command to list all Chelsio CNAs present on the system. The list displays a unique index for each CNA found.

```
C:\CHELSIO>cfut4 -l
```



iv. Delete any previous version of Option ROM flashed onto the CNA:

```
C:\CHELSIO>cfut4 -d <idx> -xb
```

Here, `idx` is the CNA index found in step iii (0 in this case)

```
C:\CHELSIO>cfut4 -d 0 -xb

 Chelsio T4/T5 Flash Utility v1.5

 Erasing serial flash sector(s) ... Done
 Reboot machine for changes to take effect
```

v. Delete any previous firmware using the following command:

```
C:\CHELSIO>cfut4 -d <idx> -xh -xf
```

```
C:\CHELSIO>cfut4 -d 0 -xh -xf

 Chelsio T4/T5 Flash Utility v1.5

 Erasing serial flash sector(s) ... Done
 Erasing serial flash sector(s) ... Done
 Reboot machine for changes to take effect
C:\CHELSIO>_
```

vi. Delete any previous Option ROM settings:

```
C:\CHELSIO>cfut4 -d <idx> -xc
```

```
C:\CHELSIO>cfut4 -d 0 -xc

 Chelsio T4/T5 Flash Utility v1.5

 Erasing serial flash sector(s) ... Done
 Reboot machine for changes to take effect
```

vii. Run the following command to flash the appropriate firmware.

```
C:\CHELSIO>cfut4 -d <idx> -uf <firmware_file>.bin
```

Here, `firmware_file` is the firmware image file present in the *CHELSIO* directory.

```
C:\CHELSIO>cfut4 -d 0 -uf T5FW-1~1.BIN

 Chelsio T4/T5 Flash Utility v1.5

 Erasing serial flash sector(s) ... Done
 Writing Image at Base 00080000 ... Done
 Writing Image at Base 00088000 ... Done
 Writing Image at Base 00090000 ... Done
 Writing Image at Base 00098000 ... Done
 Writing Image at Base 000a0000 ... Done
 Writing Image at Base 000a8000 ... Done
 Writing Image at Base 000b0000 ... Done
 Writing Image at Base 000b8000 ... Done
 Writing Image at Base 000c0000 ... Done
 Writing Image at Base 000c8000 ... Done
 Writing Image at Base 000d0000 ... Done
 Writing Image at Base 000d8000 ... Done
 Writing Image at Base 000e0000 ... Done
 Writing Image at Base 000e8000 ... Done
 Writing Image at Base 000f0000 ... Done
 Reboot machine for changes to take effect
```

viii. Flash the unified option ROM onto the Chelsio CNA using the following command:

```
C:\CHELSIO>cfut4 -d <idx> -ub cubt4.bin
```

Here, `cubt4.bin` is the unified option ROM image file present in the *CHELSIO* directory.

```
C:\CHELSIO>cfut4 -d 0 -ub cubt4.bin

 Chelsio T4/T5 Flash Utility v1.5

 Erasing serial flash sector(s) ... Done
 Writing Image at Base 00000000 ... Done
 Writing Image at Base 00008000 ... Done
 Writing Image at Base 00010000 ... Done
 Writing Image at Base 00018000 ... Done
 Writing Image at Base 00020000 ... Done
 Writing Image at Base 00028000 ... Done
 Writing Image at Base 00030000 ... Done
 Writing Image at Base 00038000 ... Done
 Writing Image at Base 00040000 ... Done
 Writing Image at Base 00048000 ... Done
 Writing Image at Base 00050000 ... Done
 Writing Image at Base 00058000 ... Done
 Writing Image at Base 00060000 ... Done
 Writing Image at Base 00068000 ... Done
 Erasing serial flash sector(s) ... Done
 Writing Image at Base 00070000 ... Done
 Reboot machine for changes to take effect
```

ix. Flash the default boot configuration file.

```
C:\CHELSIO>cfut4 -d <idx> -uc bootcfg
```

```
C:\CHELSIO>cfut4 -d 0 -uc bootcfg

 Chelsio T4/T5 Flash Utility v1.5

 Erasing serial flash sector(s) ... Done
 Updating the configuration in flash
 Writing Image at Base 00070000 ... Done
 Updating the configuration in flash succeeded
```

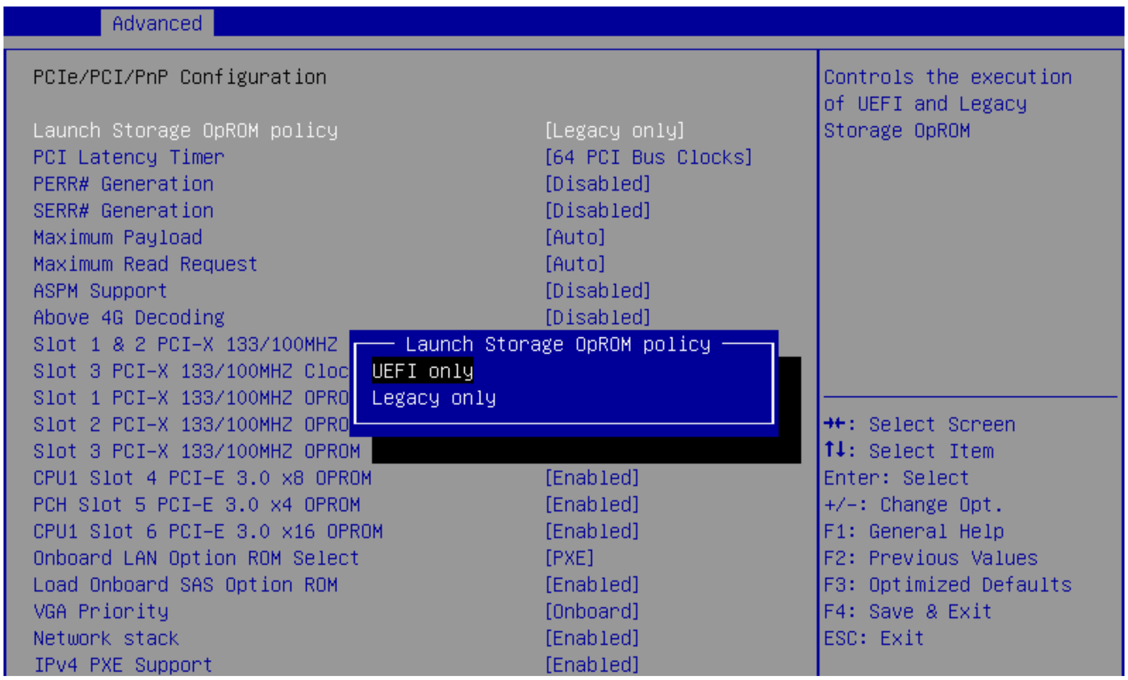
x. Reboot the system for changes to take effect.

2.3. uEFI

i. Reboot the system and go into BIOS setup.

ii. Disable Secure Boot.

iii. Configure the system having Chelsio CNA to boot in uEFI mode.



Note *For Supermicro systems, enable* ***Network Stack*** as well before proceeding.

iv. Boot to EFI Shell.

```
EFI Shell version 2.31 [4.654]
Current running mode 1.1.2
Device mapping table
  fs0  :Removable HardDisk - Alias hd83b0f0b blk0
        PciRoot(0x0)/Pci(0x1d,0x0)/USB(0x1,0x0)/USB(0x5,0x0)/HD(1,MBR,0x0fdb738d,0x800,0x78b800)
  blk0 :Removable HardDisk - Alias hd83b0f0b fs0
        PciRoot(0x0)/Pci(0x1d,0x0)/USB(0x1,0x0)/USB(0x5,0x0)/HD(1,MBR,0x0fdb738d,0x800,0x78b800)
  blk1 :HardDisk - Alias (null)
        PciRoot(0x0)/Pci(0x1f,0x2)/Sata(0x0,0x0)/HD(1,MBR,0x00092b0c,0x3f,0x9c25fe)
  blk2 :HardDisk - Alias (null)
        PciRoot(0x0)/Pci(0x1f,0x2)/Sata(0x0,0x0)/HD(2,MBR,0x00092b0c,0x9c263d,0x88b8fdc)
  blk3 :HardDisk - Alias (null)
        PciRoot(0x0)/Pci(0x1f,0x2)/Sata(0x0,0x0)/HD(3,MBR,0x00000000,0x927be19,0x14019e7)
  blk4 :HardDisk - Alias (null)
        PciRoot(0x0)/Pci(0x1f,0x2)/Sata(0x0,0x0)/HD(4,MBR,0x00000000,0xa67d83f,0x13fe849)
  blk5 :BlockDevice - Alias (null)
        PciRoot(0x0)/Pci(0x1f,0x2)/Sata(0x0,0x0)
  blk6 :Removable BlockDevice - Alias (null)
        PciRoot(0x0)/Pci(0x1d,0x0)/USB(0x1,0x0)/USB(0x5,0x0)

Press ESC in 1 seconds to skip startup.nsh, any other key to continue.
Shell> _
```

v. Issue command `drivers` to determine if Chelsio uEFI driver is loaded. If the driver is loaded (as shown in the image below), continue to step (iv)

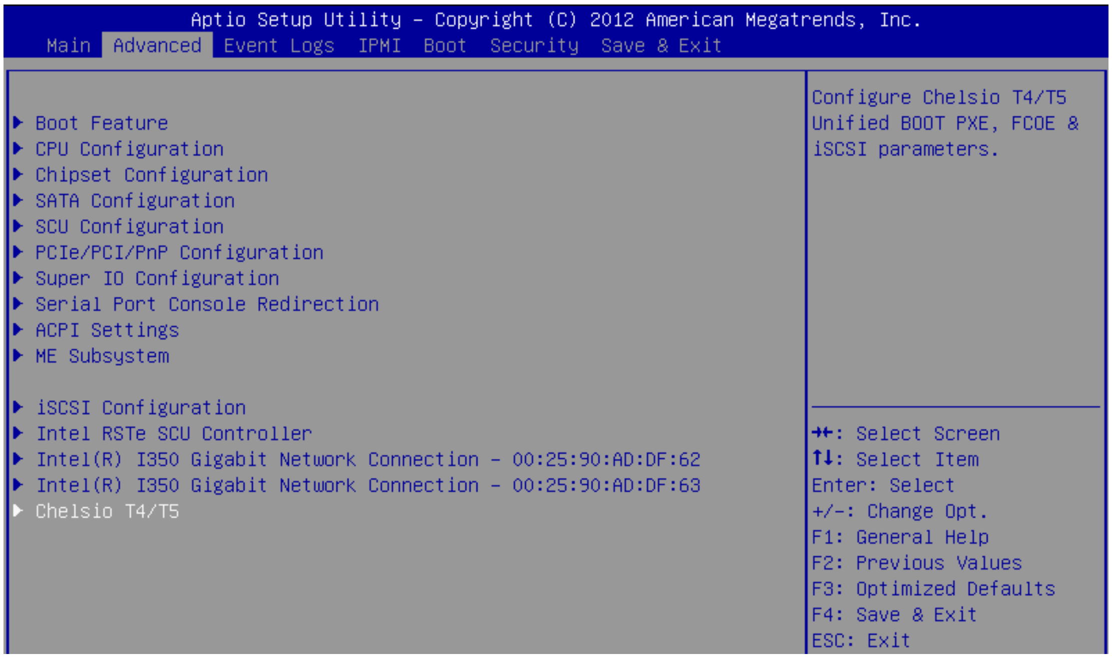
```
64 00000024 B - -  1  1 BIOS[INT10] Video Driver        CsmVideo
65 00000010 ? - -  -  - <UNKNOWN>                       <UNKNOWN>
68 00000001 B - -  1  1 AMI AHCI BUS Driver             AHCI
6C 00000010 B - -  3  3 <UNKNOWN>                       Terminal
6D 00000010 B - -  1  1 <UNKNOWN>                       Terminal
79 0000000A ? - -  -  - SCSI Bus Driver                 ScsiBus
7A 0000000A ? - -  -  - Scsi Disk Driver                ScsiDisk
7F 05061000 B X X  2  2 Intel(R) PRO/1000 5.6.10 PCI-E  IntelGigabitLanx64
80 0000000A D - -  2  - iSCSI Driver                    IScsiDxe
85 00000010 D - -  1  - <UNKNOWN>                       BIOSBLKIO
BD 0000008A D - -  2  - AMI USB Driver                  UHCD
BF 0000008A B - -  2  5 USB bus                         UHCD
C0 00000001 D - -  2  - USB Hid driver                  UHCD
C1 00000001 D - -  1  - USB Mass Storage driver         UHCD
C2 00000001 ? - -  -  - AMI USB CCID driver             UHCD
E3 00000010 D - -  7  - <UNKNOWN>                       CORE_DXE
E4 00000010 D - -  1  - <UNKNOWN>                       CORE_DXE
E5 00000010 B - -  6  6 <UNKNOWN>                       CORE_DXE
E7 00000010 B - -  2  5 <UNKNOWN>                       CORE_DXE
E8 00000010 D - -  1  - AMI PS/2 Driver                 CORE_DXE
E9 00000010 ? - -  -  - AMI Floppy Driver               CORE_DXE
EA 00000001 ? - -  -  - AMI IDE BUS Driver              CORE_DXE
F8 0100004B B X X  2  - Chelsio Unified Driver          Offset(0x3034,0x1a

Shell> drivers_
```

If the driver is not loaded, load the uEFI driver (*ChelsioUD.efi*) found in the CHELSIO directory, and try again.

```
fs0:\CHELSIO> load ChelsioUD.efi
load: Image fs0:\CHELSIO\ChelsioUD.efi loaded at 7F2BA000 - Success
```

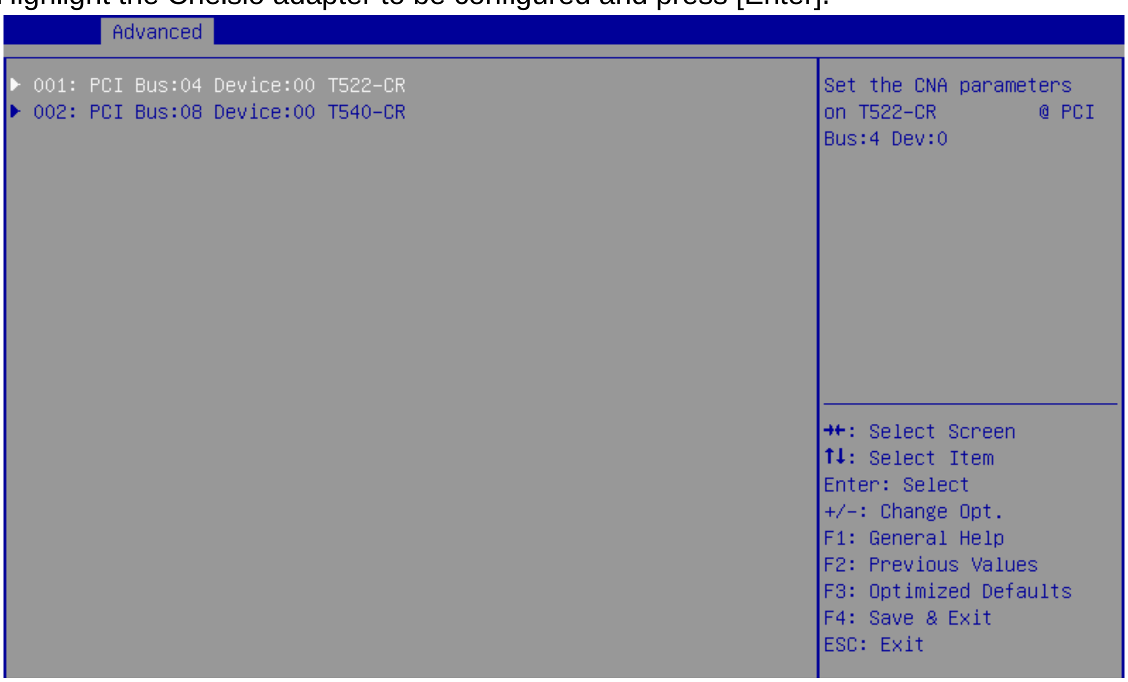
vi. Reboot the system and go into BIOS setup.

vii. Chelsio HII should be listed as ***Chelsio T4/T5***. Highlight it and press [Enter].

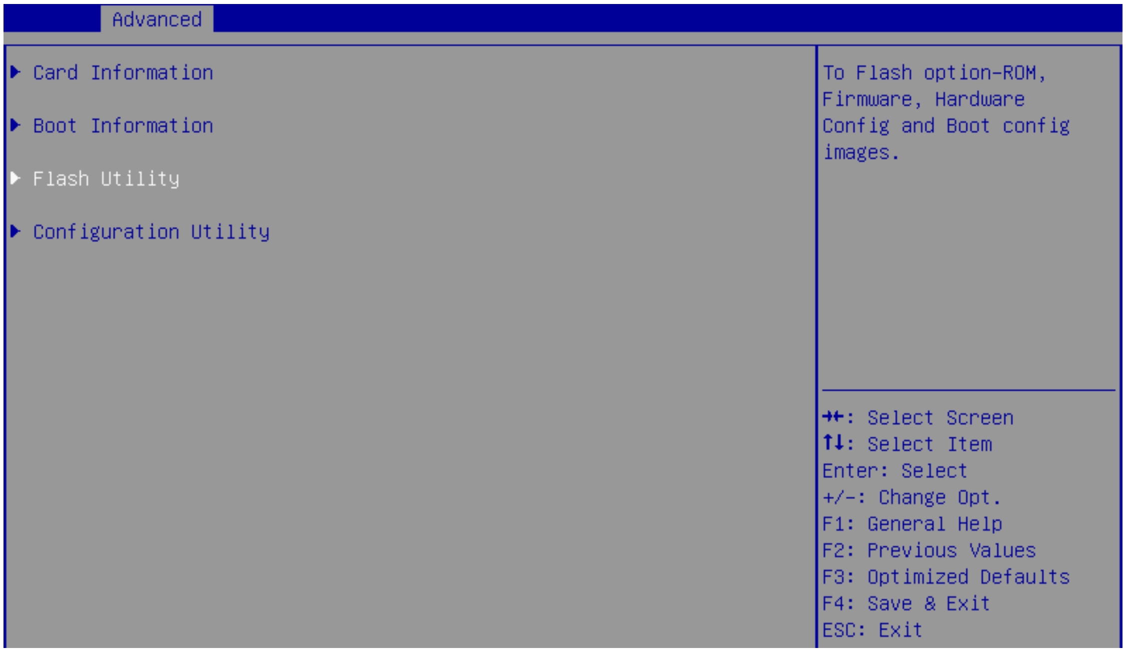


Note *If Chelsio T4/T5 is not listed, please ensure that Chelsio uEFI driver is loaded correctly as mentioned* ***here*** *in the* ***Flashing Firmware and Option ROM*** *section.*

viii. Highlight the Chelsio adapter to be configured and press [Enter].



ix. Highlight **Flash Utility** and press [Enter].



x. Erase or update firmware using the methods explained below:

- **Erase existing firmware**

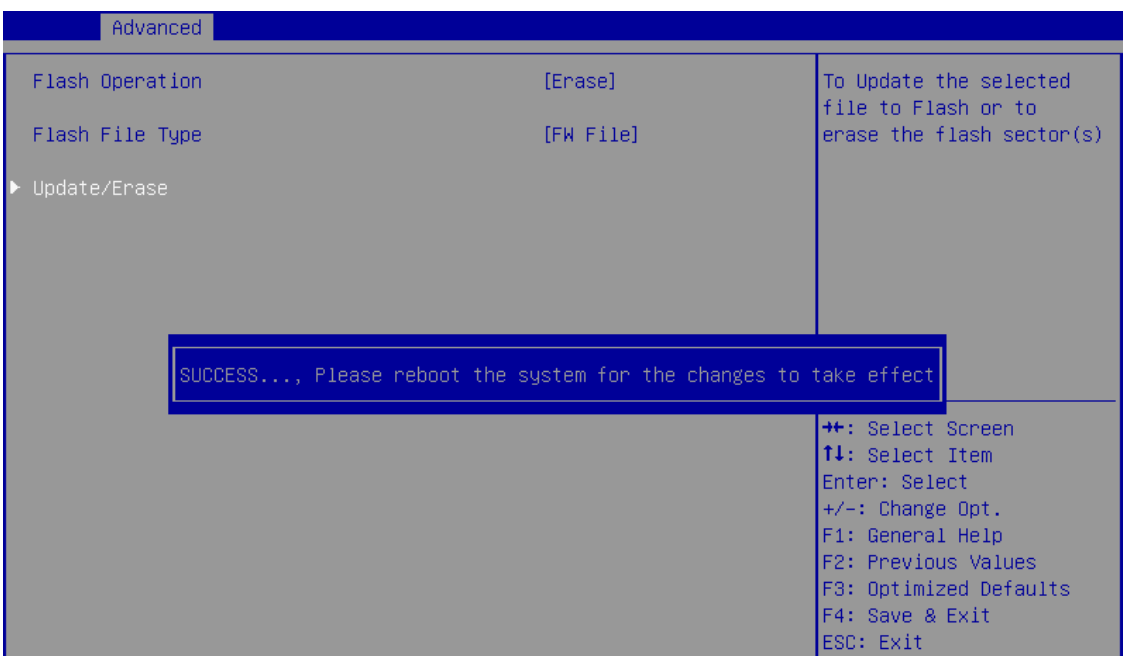
a. Select *[Erase]* as *Flash Operation*

b. Select *[FW File]* as *Flash File Type*

c. Select *Update/Erase*

d. Press [Y] to confirm.

e. Reboot system.



- **Update firmware**

a. Select *[Update]* as *Flash Operation*

b. Select *[FW File]* as *Flash File Type*

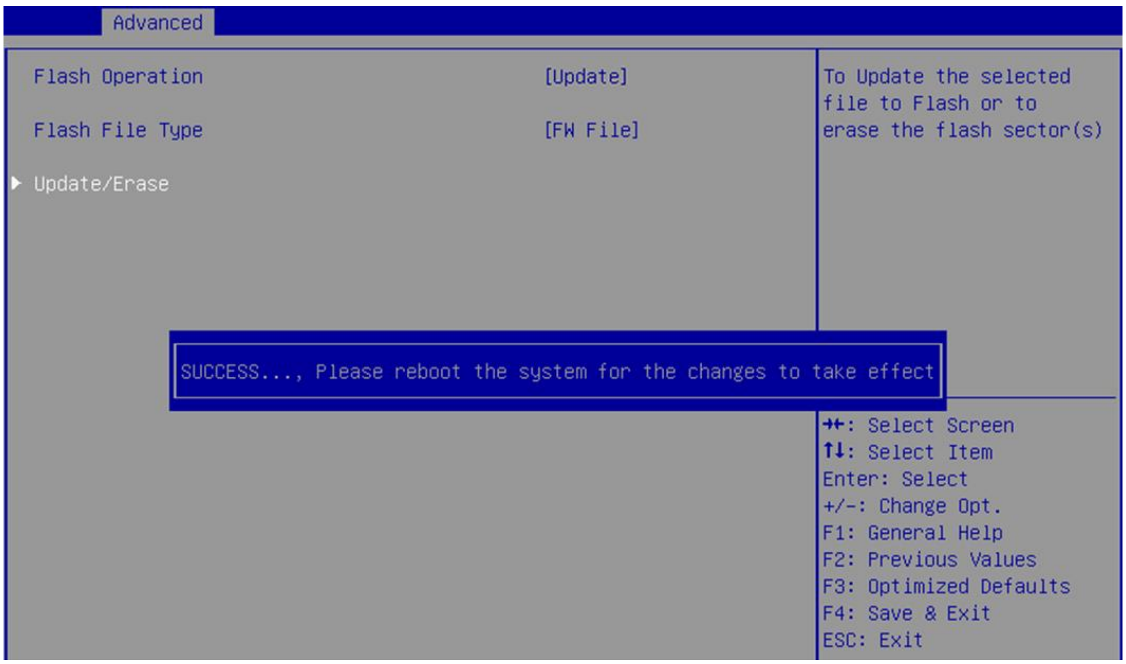
c. Enter full path to the firmware file for *Enter File Name*. For e.g.: *CHELSIO\t5fw-1.13.32.0.bin*.

d. Press [Enter]

e. Select *Update/Erase*

f. Press [Y] to confirm.

g. Reboot system



Similarly, you can use the above method to update/erase Option ROM, (T5/T4) Boot Configuration, Hardware Configuration and Phy Firmware file.

3. Configuring PXE Server

The following components are required to configure a server as PXE Server:

- DHCP Server
- TFTP Server

PXE server configuration steps for Linux can be found on following links:

- http://linux-sxs.org/internet_serving/pxeboot.html
- http://www.howtoforge.com/ubuntu_pxe_install_server

PXE server configuration steps for Windows can be found on following links:

- http://technet.microsoft.com/en-us/library/cc771670%28WS.10%29.aspx
- http://tftpd32.jounin.net/ (Use port # 67, set PXE option and provide bootable file name in settings)
- http://unattended.sourceforge.net/pxe-win2k.html

Note *Chelsio Communications does not take any responsibility regarding contents given in above mentioned links. Those are given for example purposes only.*

4. PXE boot process

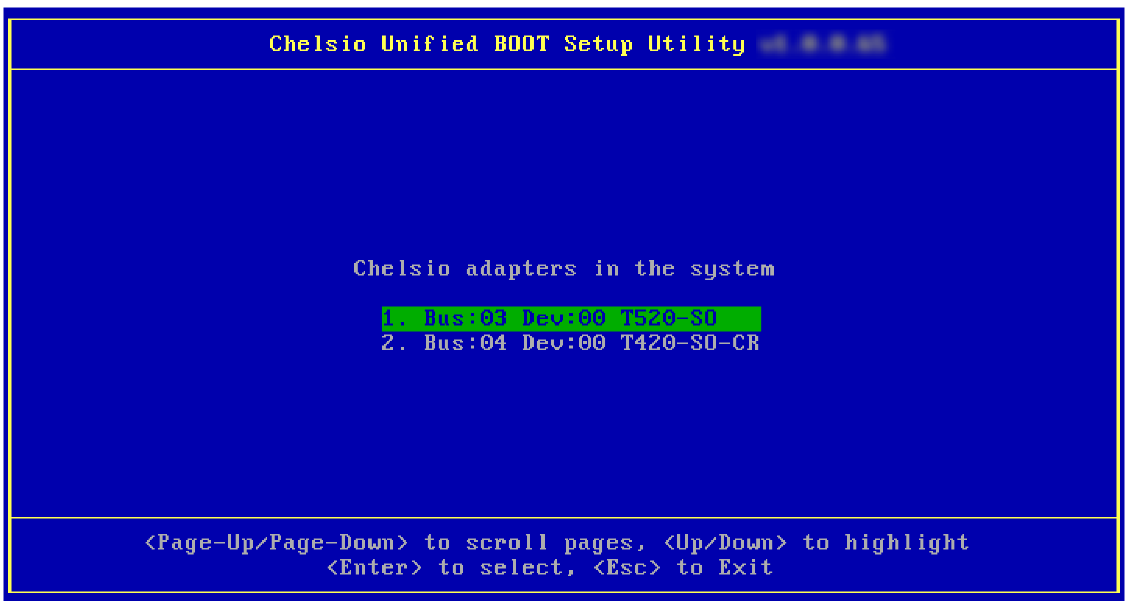
Before proceeding, please ensure that the Chelsio CNA has been flashed with the provided firmware and option ROM (See **Flashing Firmware and option ROM**).

4.1. Legacy PXE boot

i. After configuring the PXE server, make sure the PXE server works. Then reboot the client machine.

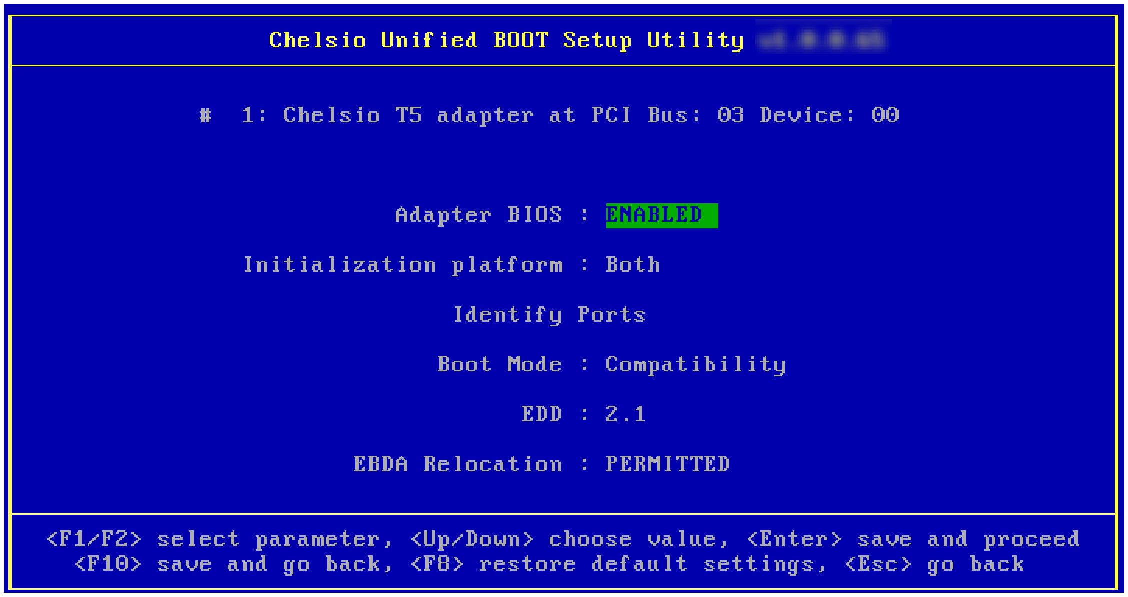
ii. Press [Alt+C] when the message *Chelsio Unified Boot BIOS vX.X.X.XX, Copyright (C) 2003-2016 Chelsio Communications Press <Alt-C> to Configure T4/T5 Card(s)*. *Press <Alt-S> to skip B*IOS appears on the screen to enter the configuration utility.

iii. The configuration utility will appear as below:



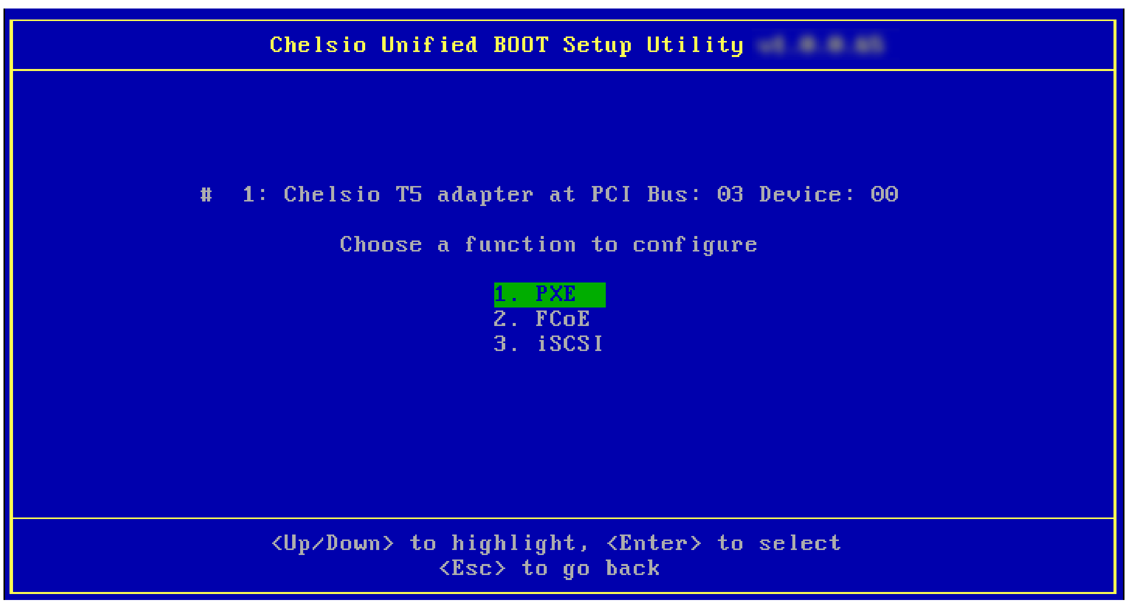
iv. Choose the CNA on which you flashed the option ROM image. Hit [Enter].

v. Enable the adapter BIOS using arrow keys if not already enabled. Hit [ENTER].

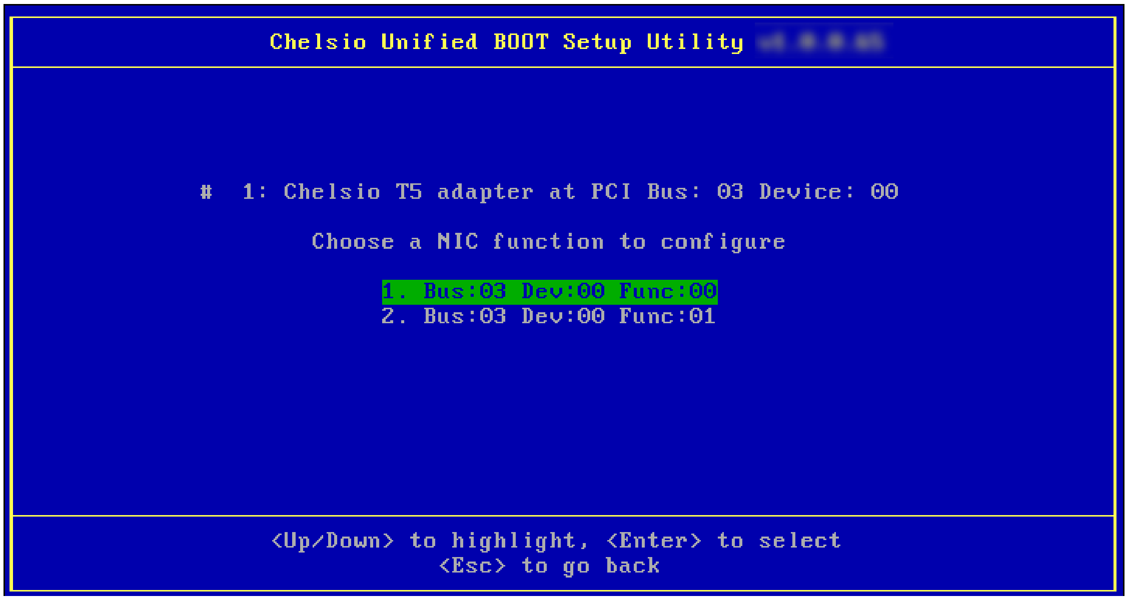


Note *Use the default values for Boot Mode, EDD and EBDA Relocation parameters, unless instructed otherwise.*

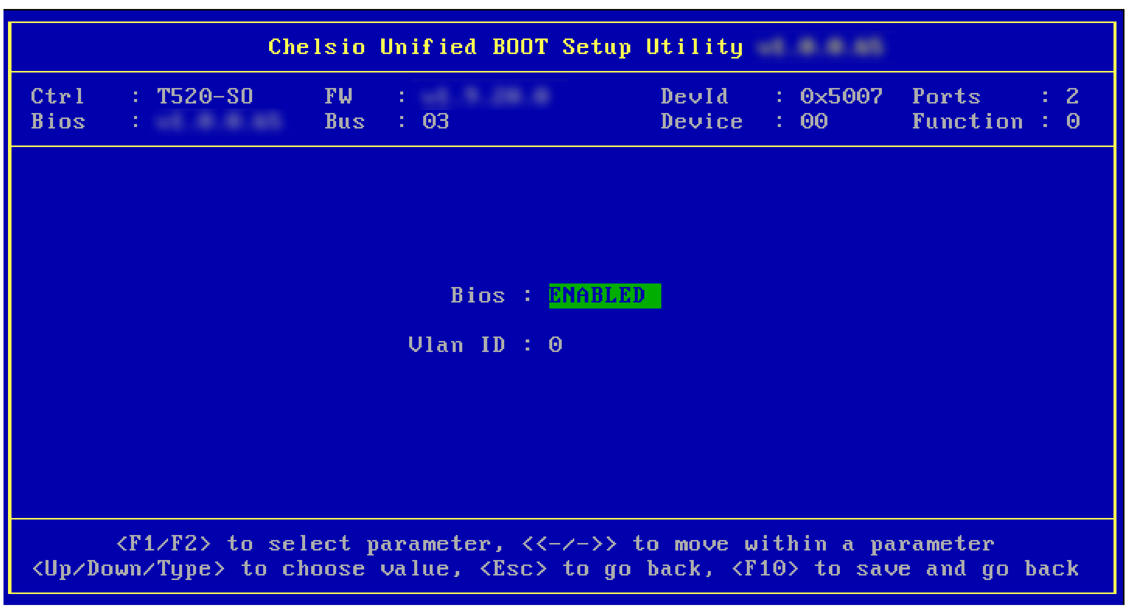
vi. Choose *PXE* from the list to configure. Hit [Enter].



vii. Use the arrow keys to highlight the appropriate function among the supported NIC functions and hit [Enter] to select.



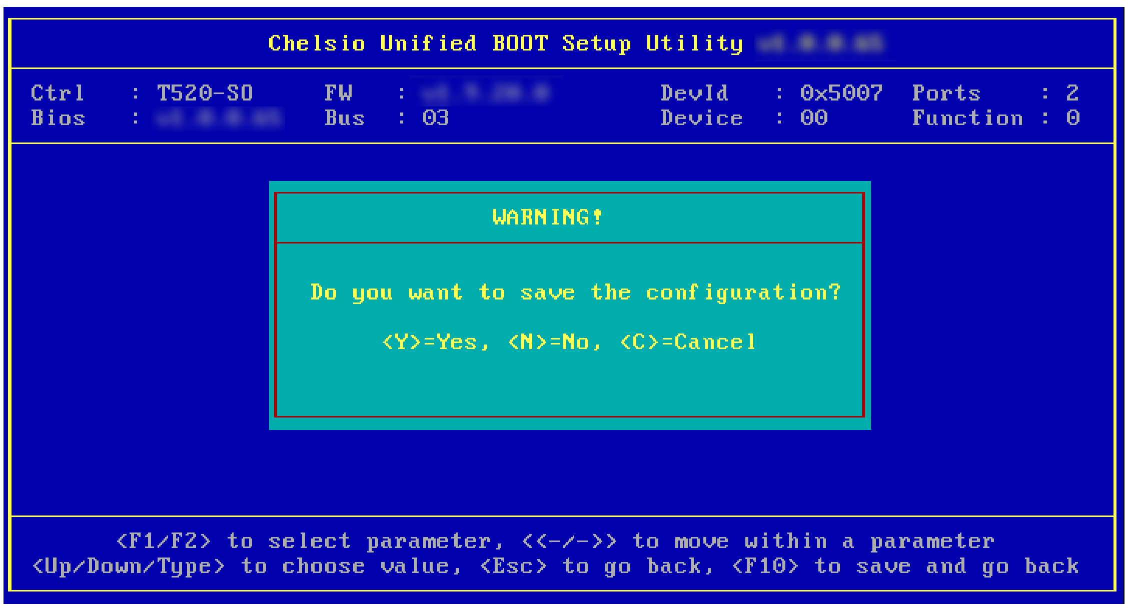
viii. Enable NIC function bios if not already enabled.



ix. Choose the boot port to try the PXE boot. It is recommended to only enable functions and ports which are going to be used. Please note that enabling NIC Func 00 will enable port 0 for PXE, enabling NIC Func 01 will enable port 1 and so on for NIC function. Please refer the table below:

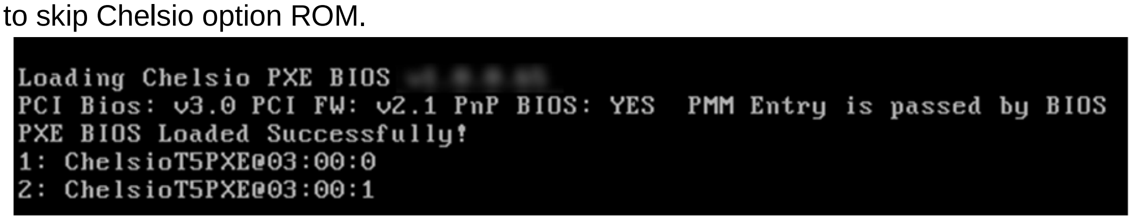
NIC Function enabled	Ports enabled
NIC Func00	00
NIC Func01	01
NIC Func02	02
NIC Func03	03

x. Hit [F10] or [Esc] and then [Y] to save configuration changes.

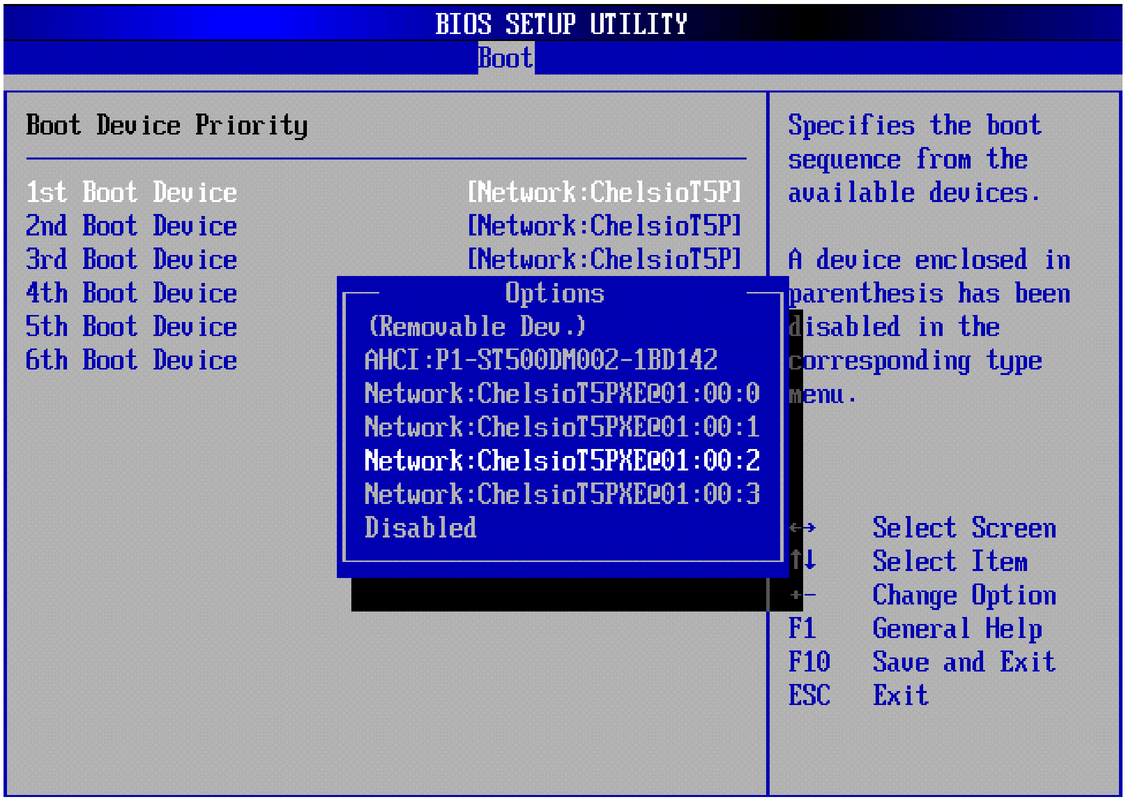


xi. Reboot the system.

xii. Allow the Chelsio option ROM to initialize and setup PXE devices. DO NOT PRESS ALT-S to skip Chelsio option ROM.



xiii. In the system setup, choose any of the Chelsio PXE devices as the first boot device.



xiv. Reboot. DO NOT PRESS ALT-S to skip Chelsio option ROM, during POST.

xv. Hit [F12] key when prompted to start PXE boot.

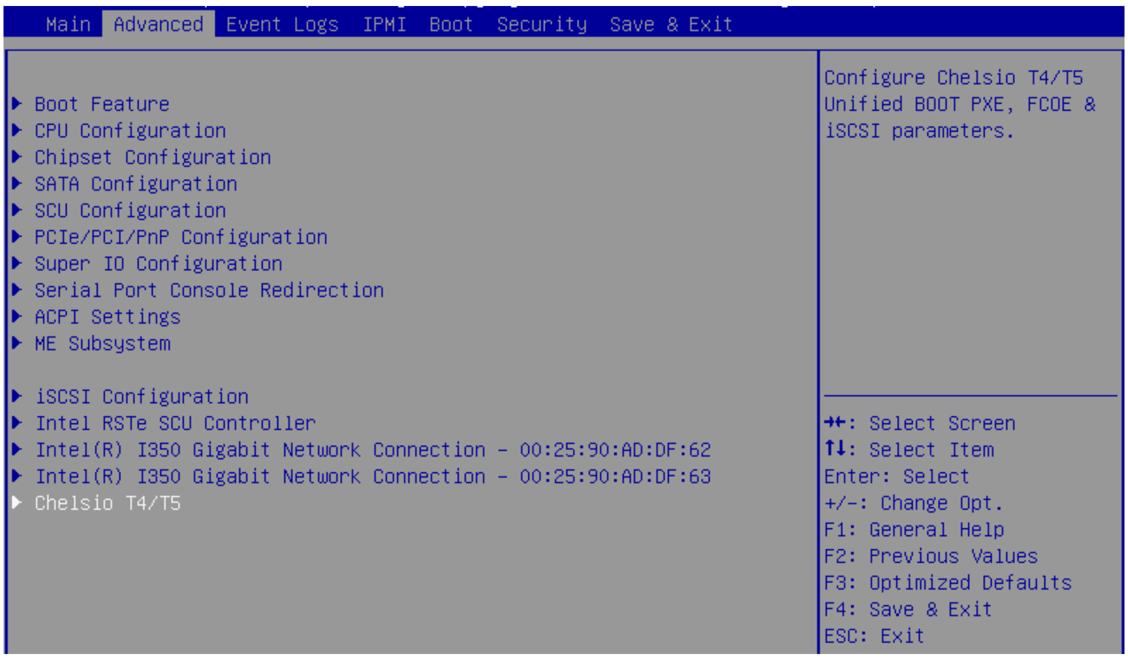
4.2. uEFI PXE Boot

Important

- *Only uEFI v2.1 and v2.3.1 supported.*
- *Any other uEFI version is NOT SUPPORTED and may render your system unusable.*

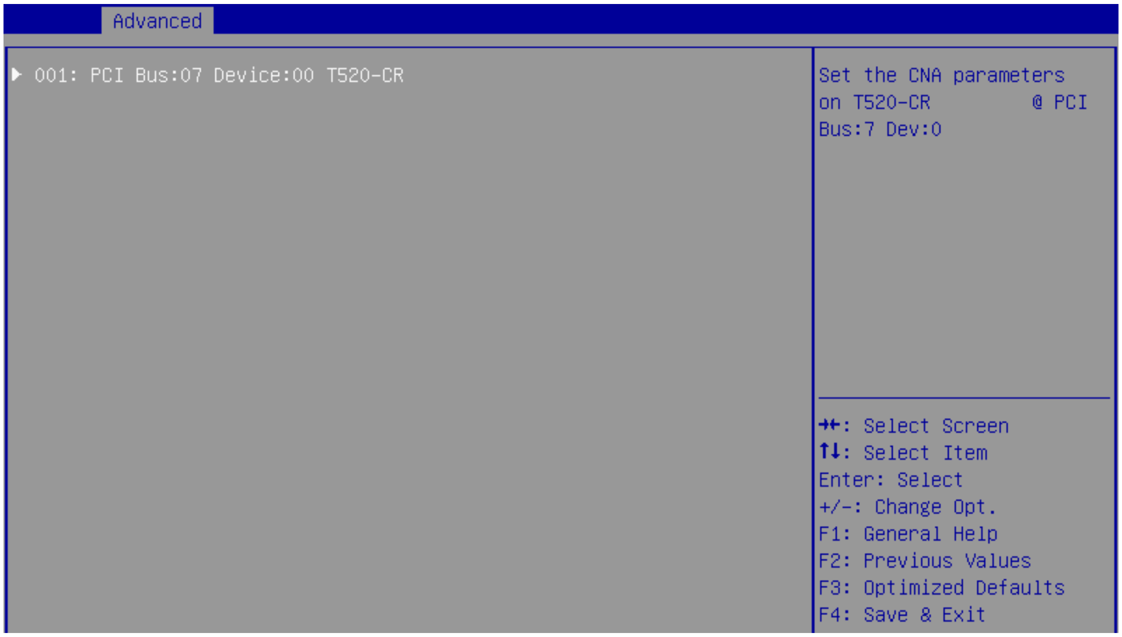
This section describes the method to configure and use Chelsio uEFI PXE interfaces.

i. Reboot the system and go into the BIOS setup.
ii. Disable Secure Boot, if not already done.
iii. Chelsio HII should be listed as **Chelsio T4/T5**. Highlight it and press [Enter].

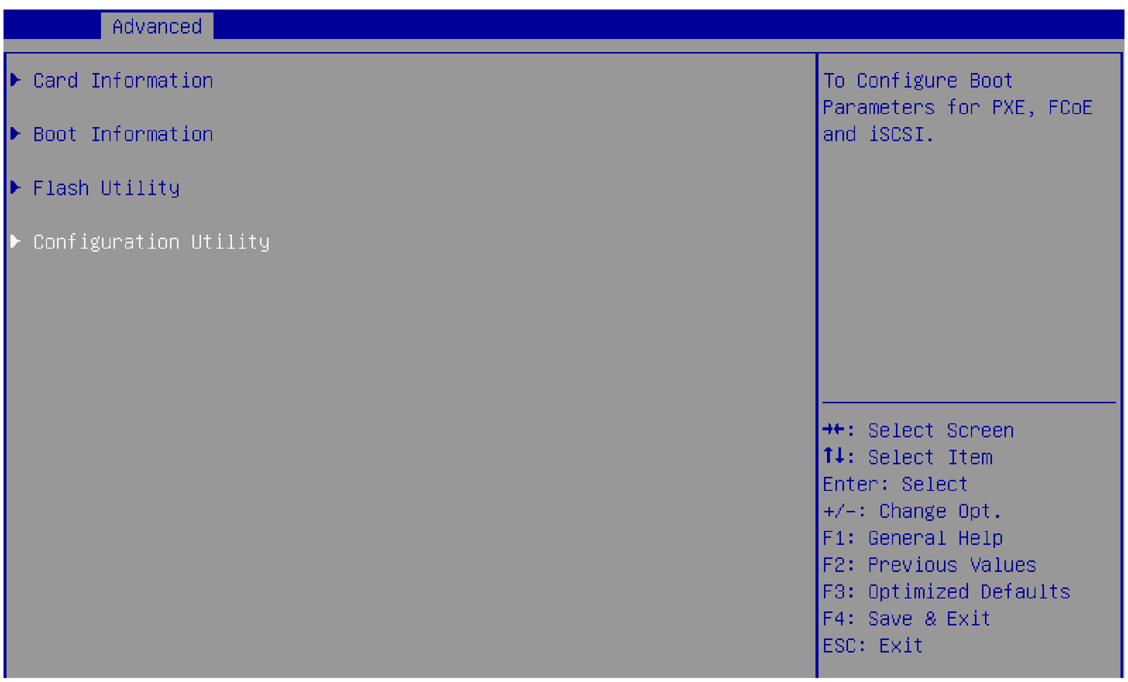


Note *If Chelsio T4/T5 is not listed, please ensure that Chelsio uEFI driver is loaded correctly as mentioned* ***here*** *in the* ***Flashing Firmware and Option ROM*** *section.*

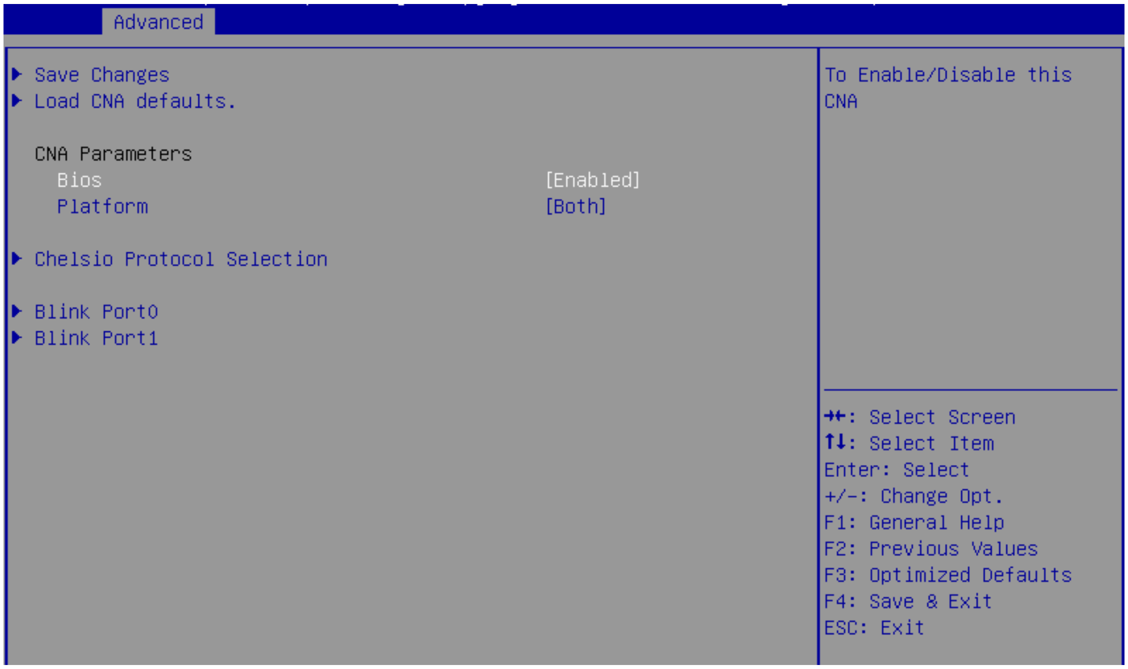
iv. Select the Chelsio adapter to be configured and press [Enter].



v. Select **Configuration Utility** and press [Enter].

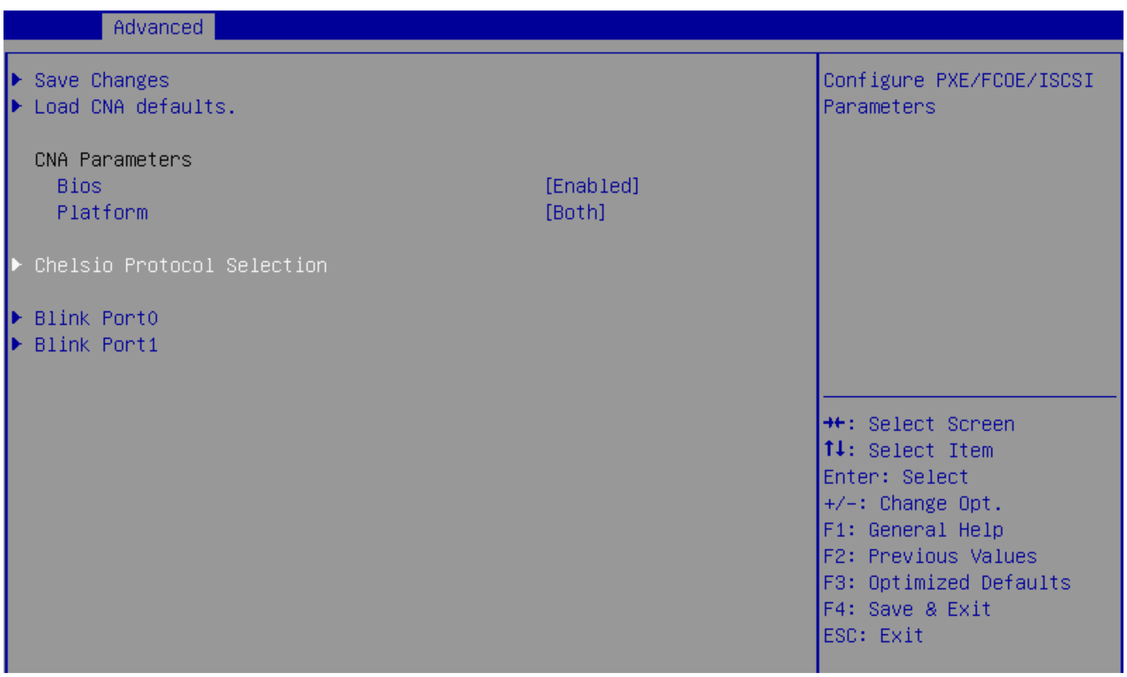


vi. Enable adapter BIOS if not already enabled.

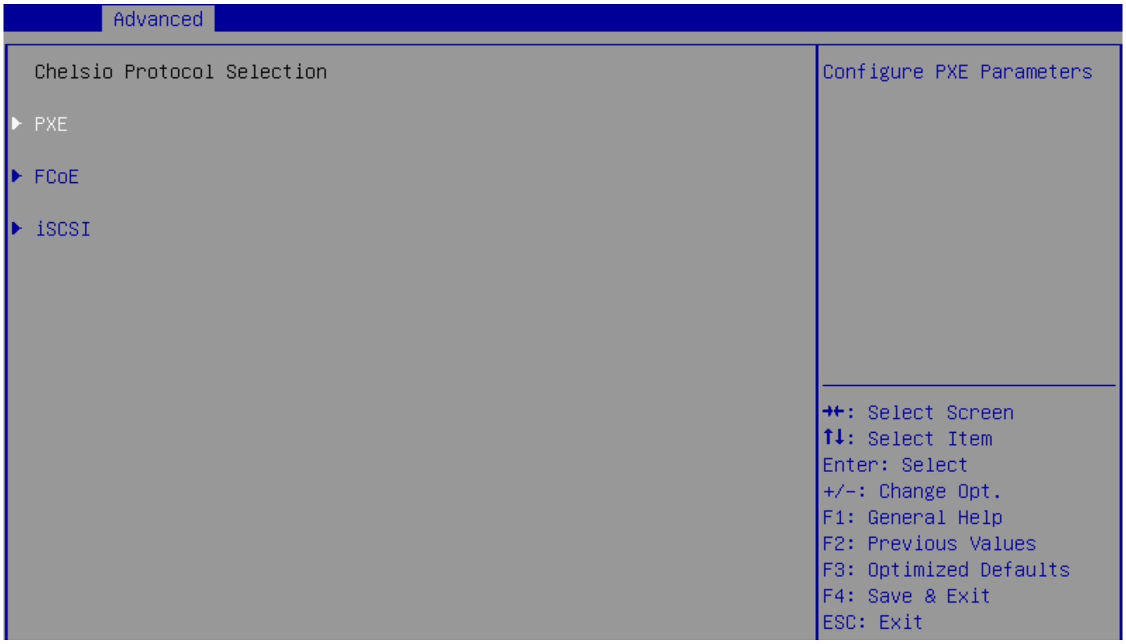


Note *It is highly recommended that you use the **Save Changes** option every time a parameter/option is changed.*

vii. Select **Chelsio Protocol Selection** and press [Enter].

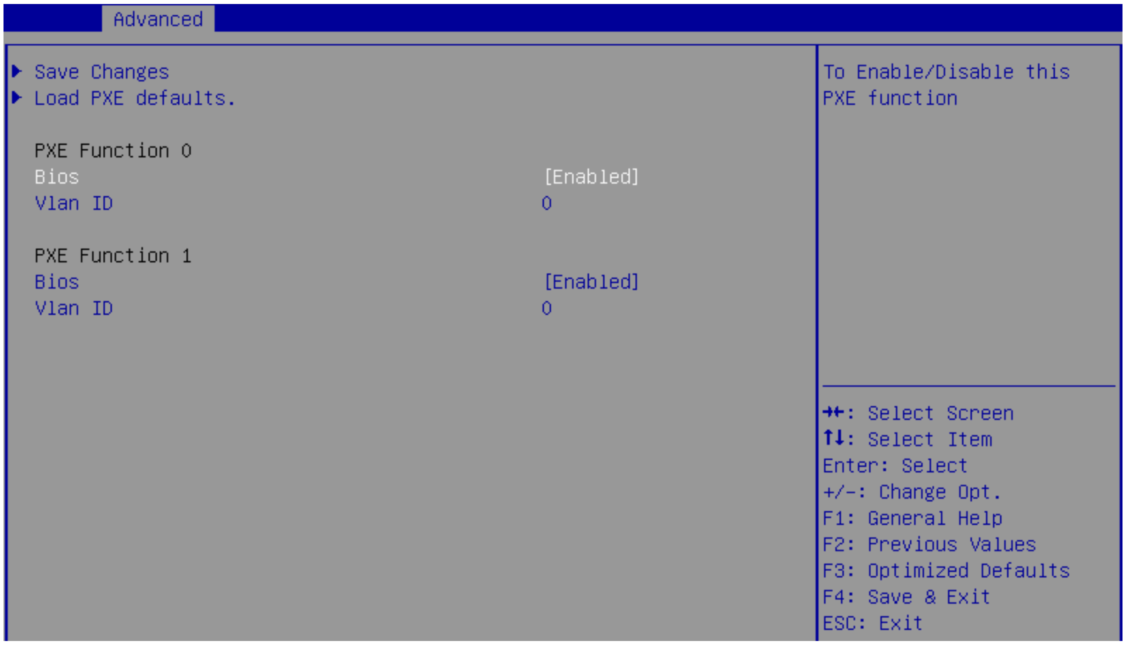


viii. Select **PXE** and press [Enter].

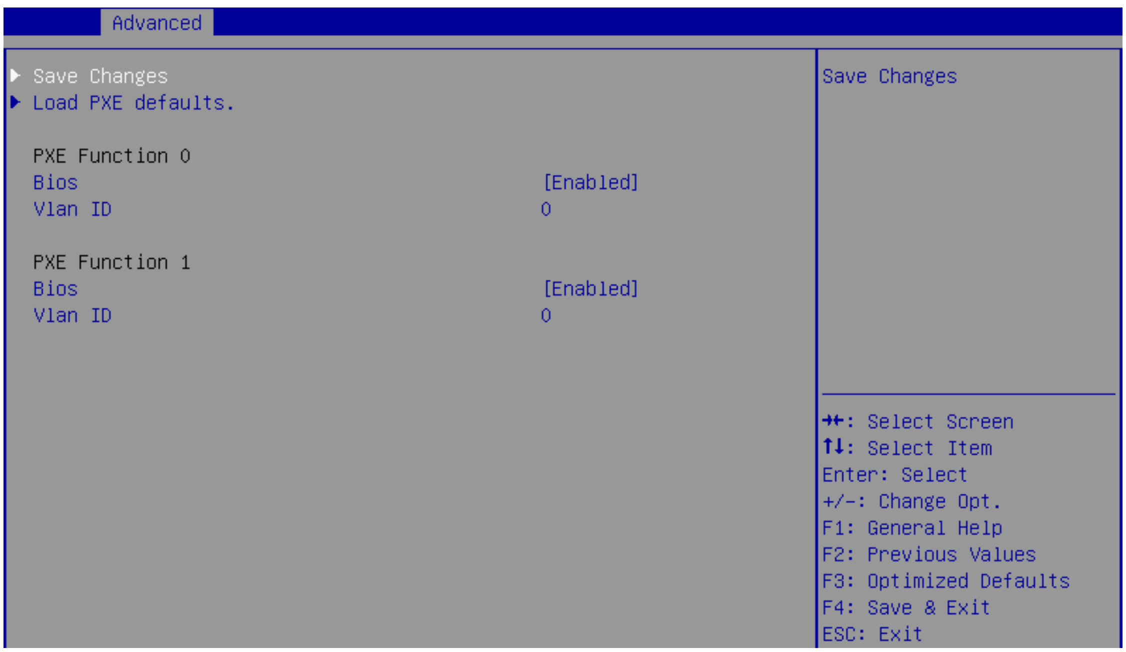


ix. Choose the boot port to try PXE boot. It is recommended to enable only those functions and ports which are going to be used. Please note that enabling PXE Function 0 will enable port 0 for PXE, enabling PXE Function 1 will enable port 1 and so on, for NIC function. Please refer the table below:

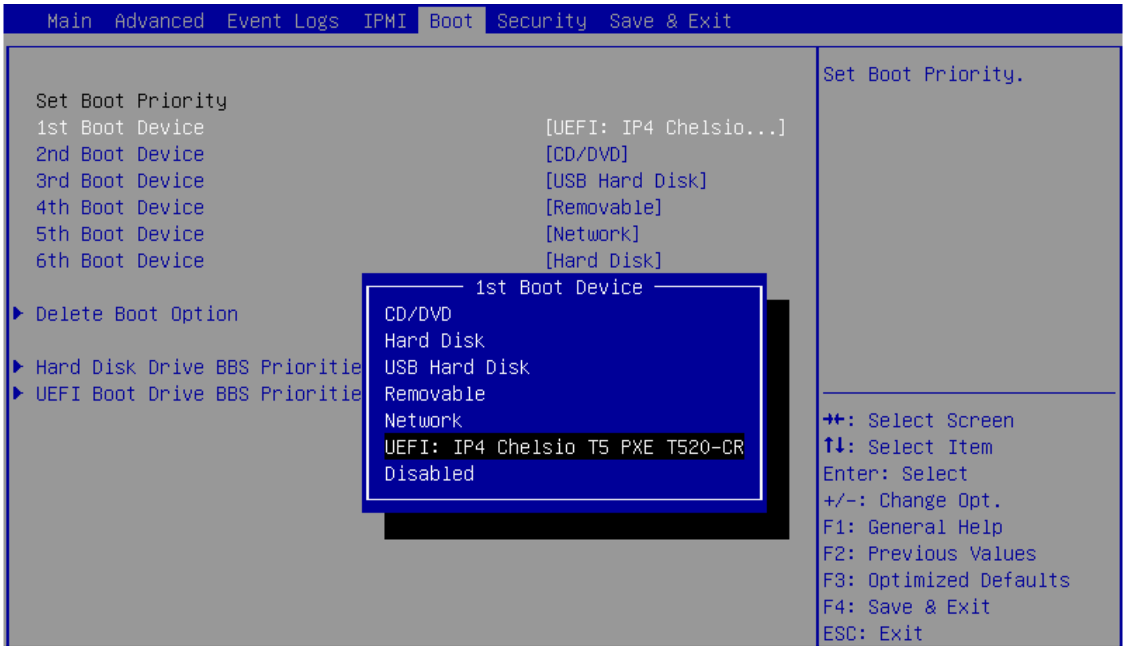
NIC Function enabled	Ports enabled
PXE Function 0	00
PXE Function 1	01
PXE Function 2	02
PXE Function 3	03



x. Select **Save Changes** and press [Enter]

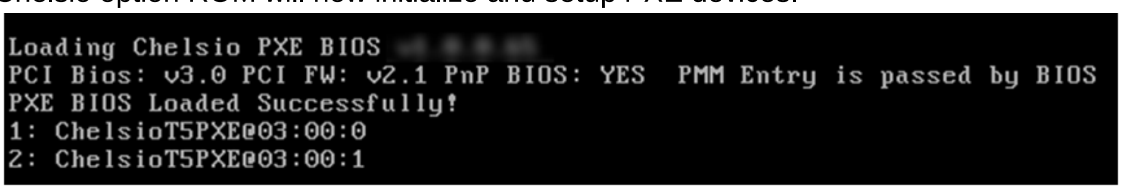


xi. Reboot the system and choose any of the available Chelsio PXE devices for PXE boot.



xii. Reboot and hit [F12] key when prompted to start PXE boot.

xiii. Chelsio option ROM will now initialize and setup PXE devices.



5. FCoE boot process

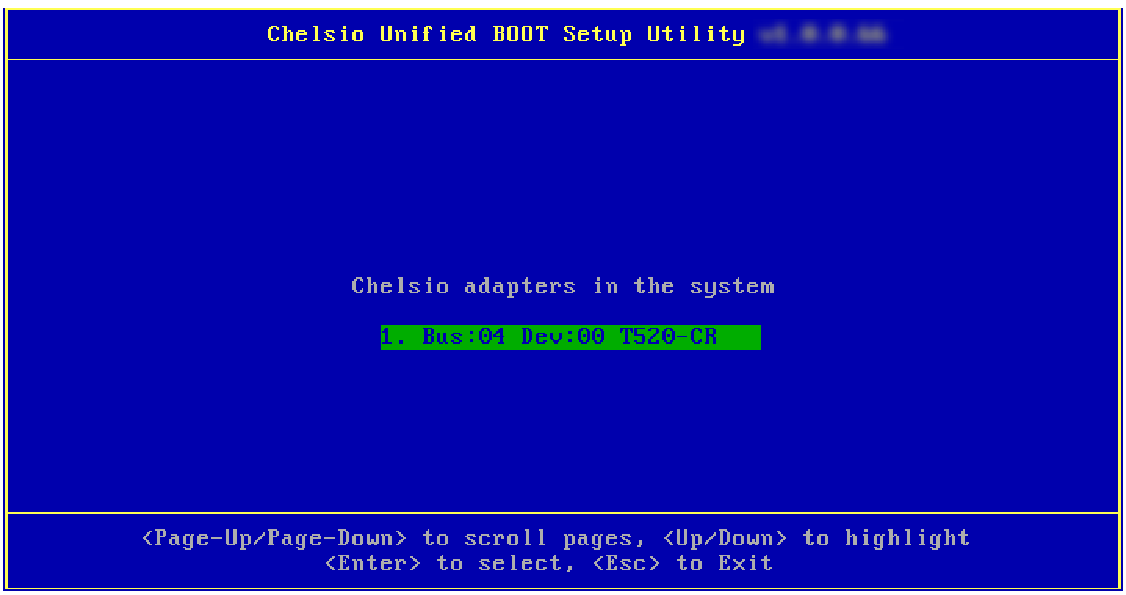
Before proceeding, please ensure that the Chelsio CNA has been flashed with the provided firmware and option ROM (See **Flashing firmware and option ROM**).

5.1. Legacy FCoE boot

i. Reboot the system.

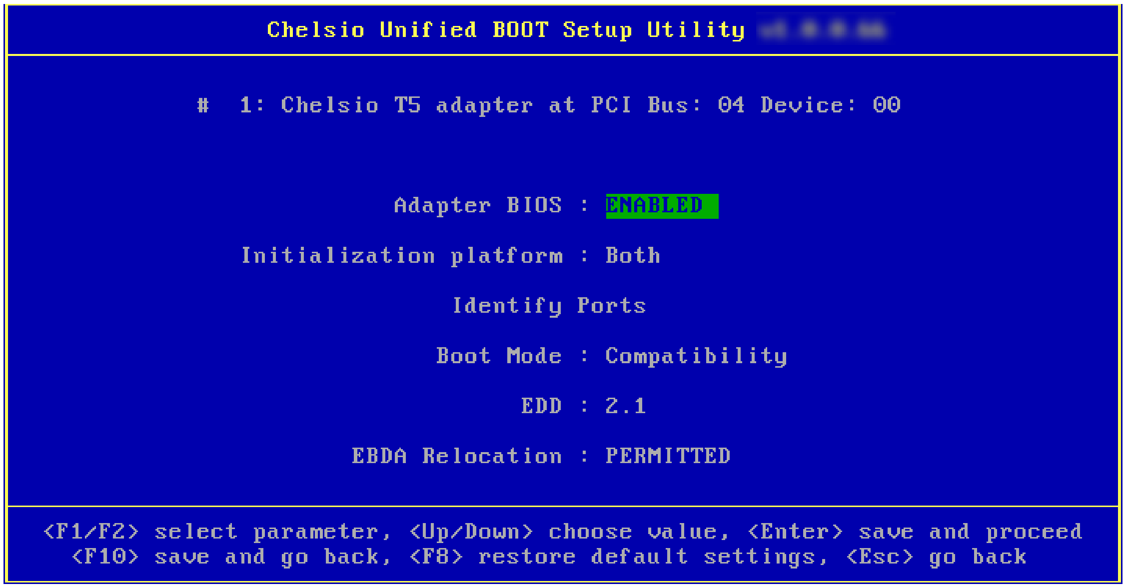
ii. Press [Alt+C] when the message "*Chelsio Unified Boot BIOS vX.X.X.XX, Copyright (C) 2003-2016 Chelsio Communications Press <Alt-C> to Configure T4/T5 Card(s)*. *Press <Alt-S> to skip B*IOS" appears on the screen to enter the configuration utility.

iii. The configuration utility will appear as below:



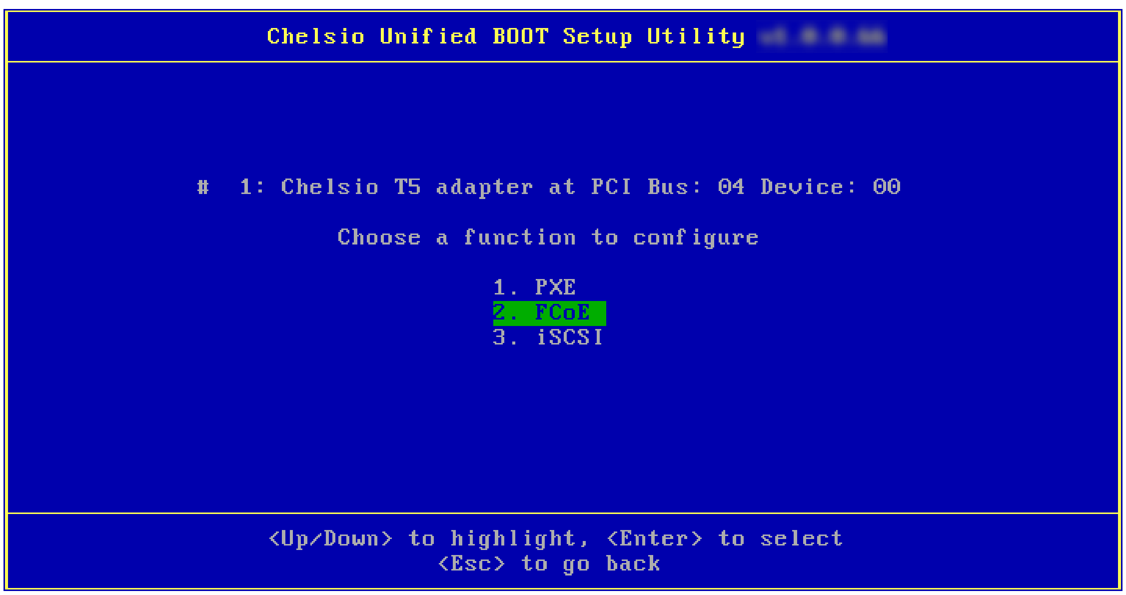
iv. Choose the CNA on which you flashed the option ROM image. Hit [Enter].

v. Enable the adapter BIOS if not already enabled. Hit [ENTER].

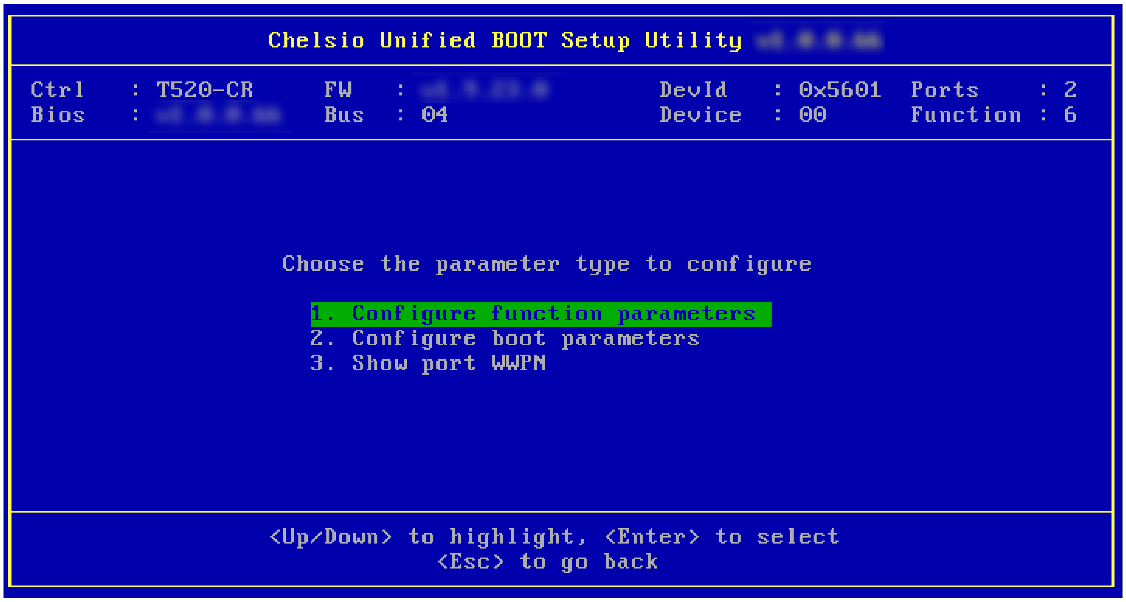


Note *Use the default values for Boot Mode, EDD and EBDA Relocation parameters, unless instructed otherwise.*

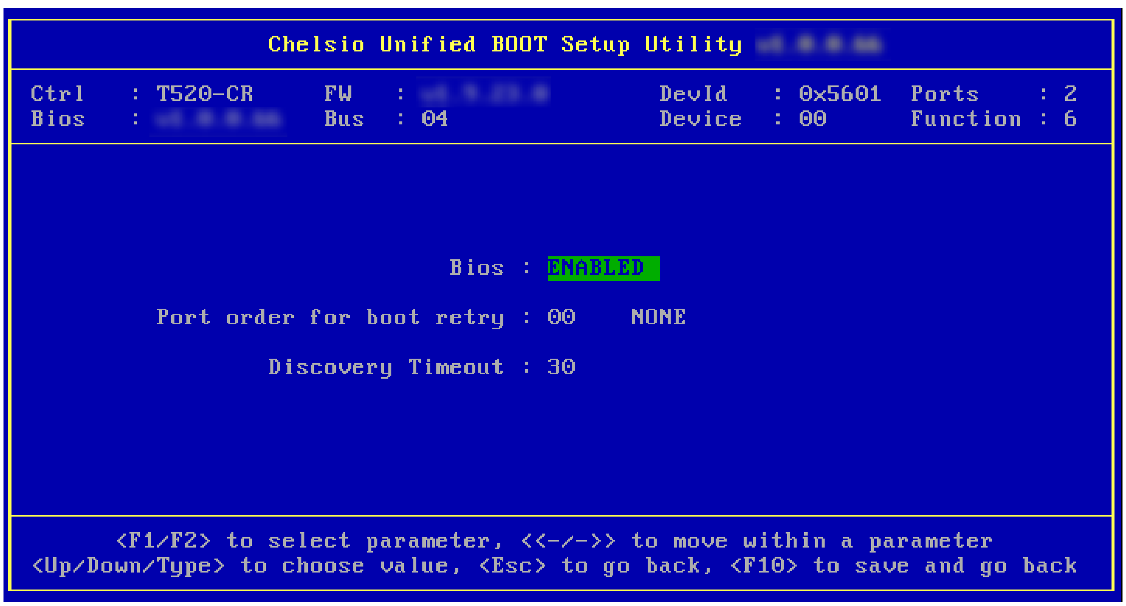
vi. Choose FCoE from the list to configure and hit [Enter].



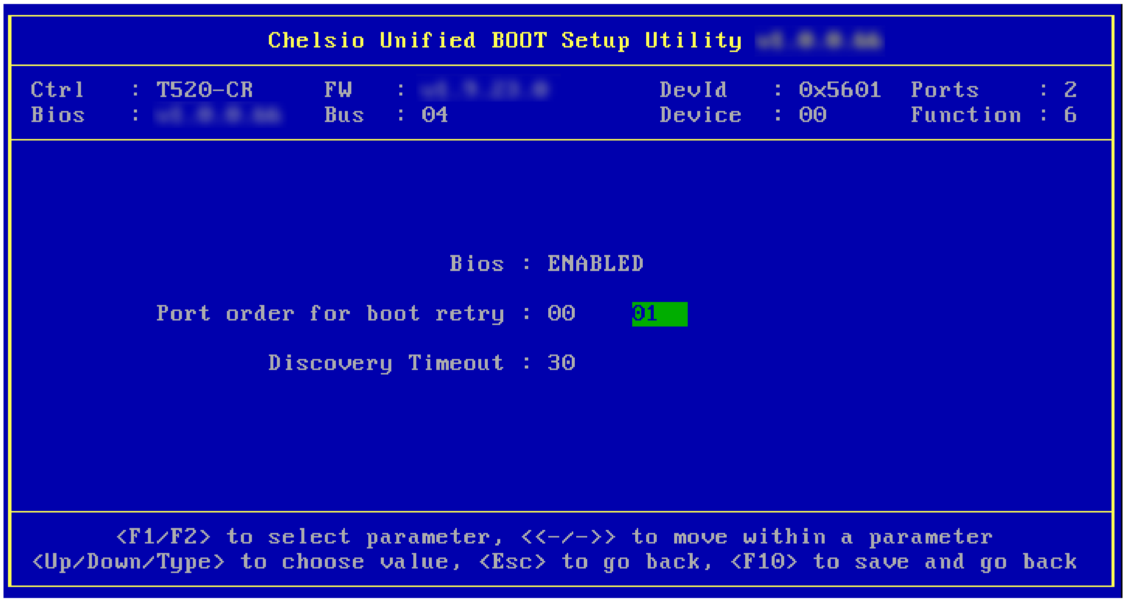
vii. Choose the first option, **Configure function parameters**, from the list of parameter type and hit [Enter].



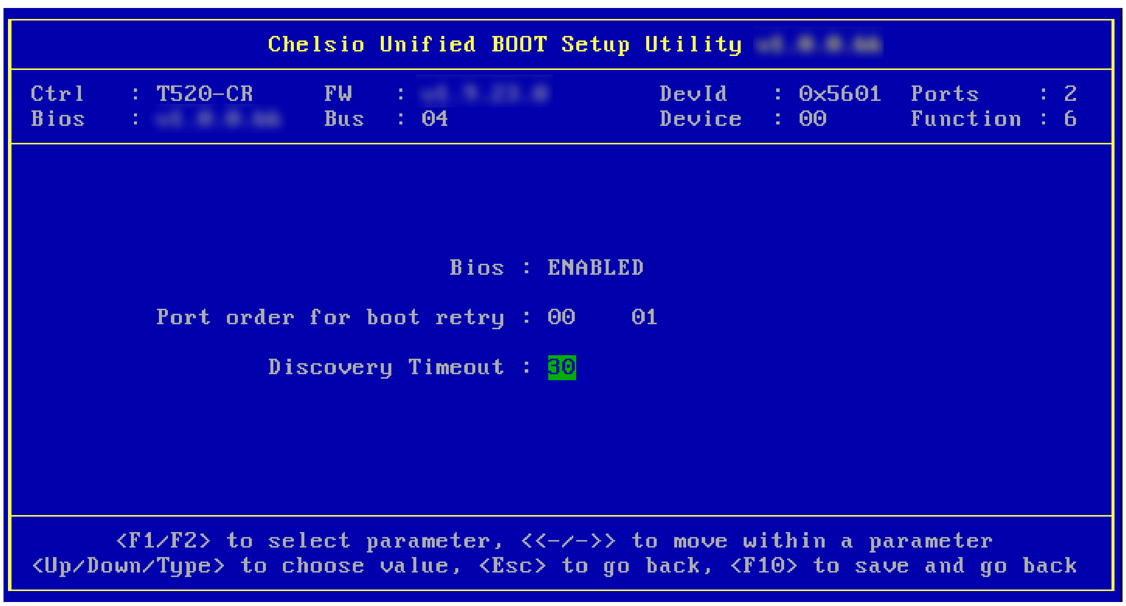
viii. Enable FCoE BIOS if not already enabled.



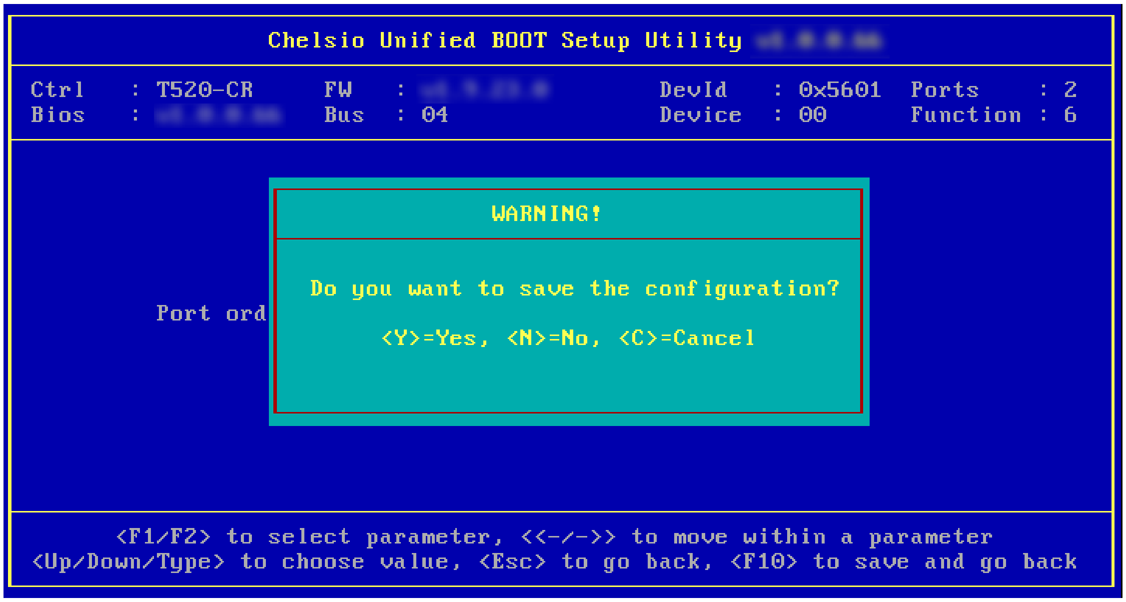
ix. Choose the order of the ports to discover FCoE targets.



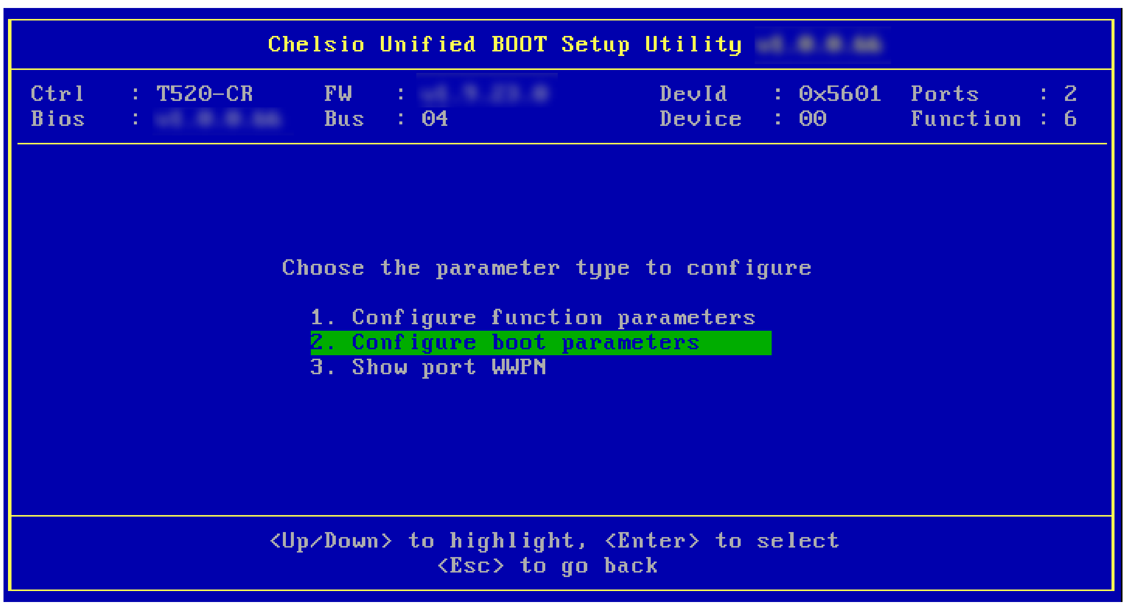
x. Set discovery timeout to a suitable value. Recommended value is >= 30.



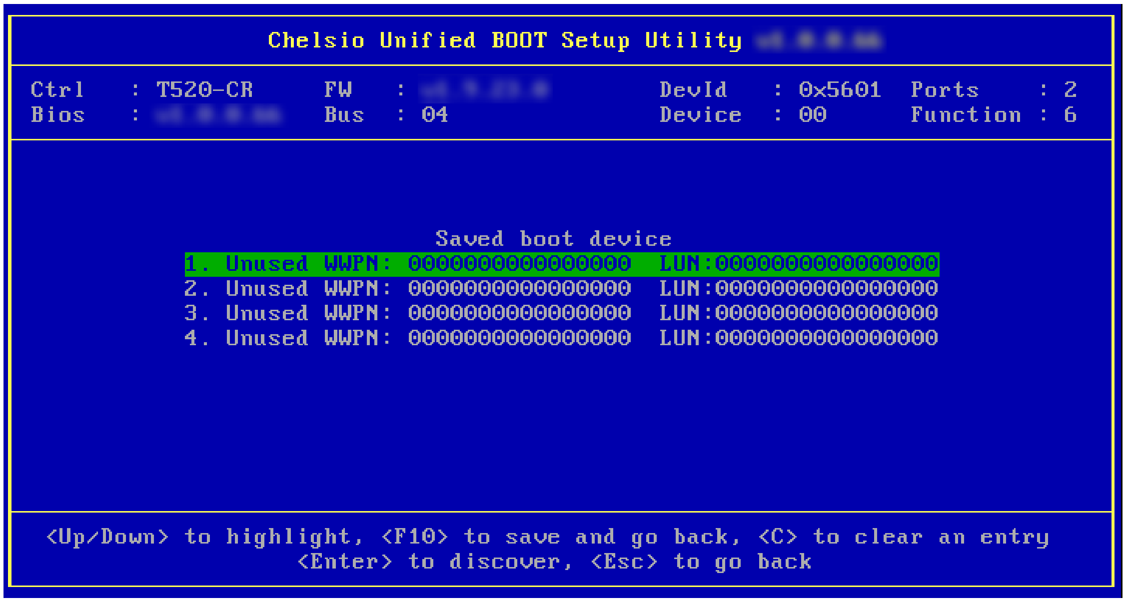
xi. Hit [F10] or [Esc] and then [Y] to save the configuration.



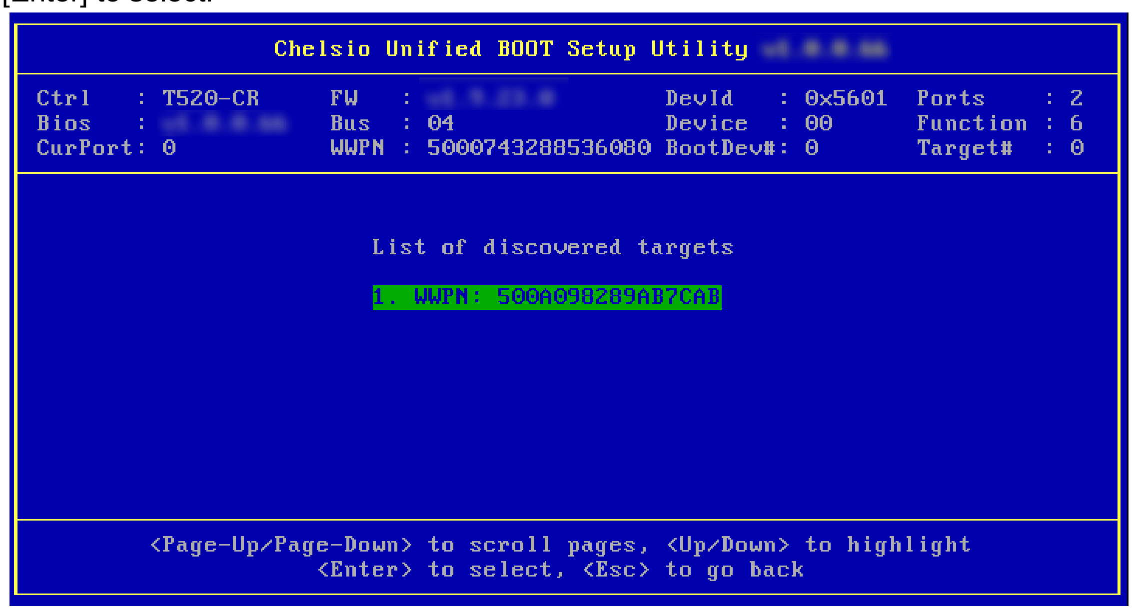
xii. Choose **Configure boot parameters**.



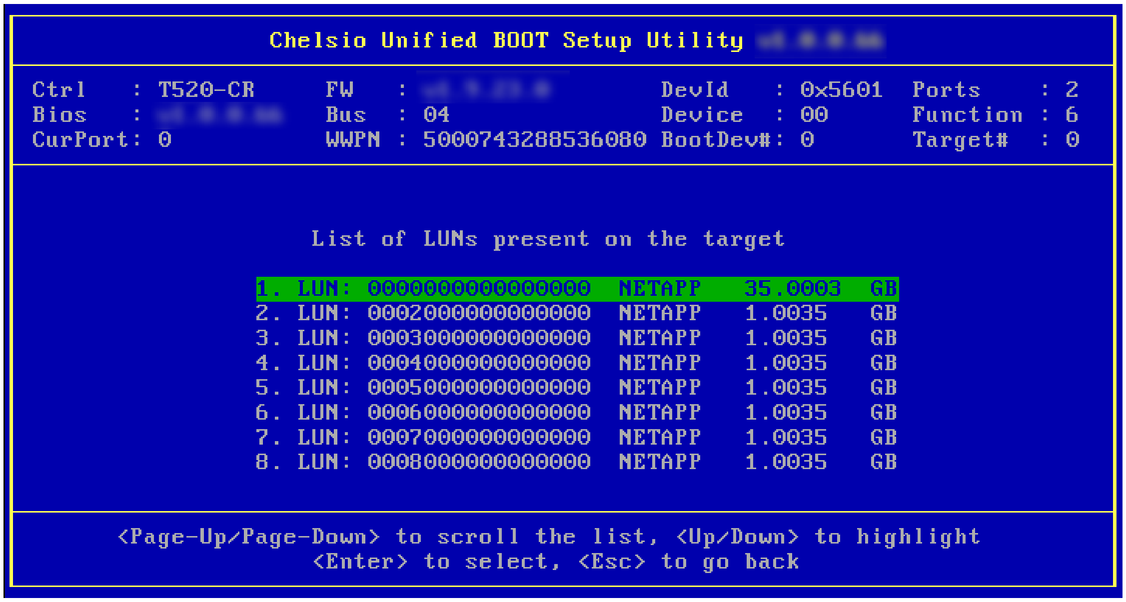
xiii. Select the first boot device and hit [Enter] to discover FC/FCoE targets connected to the switch. Wait till all reachable targets are discovered.

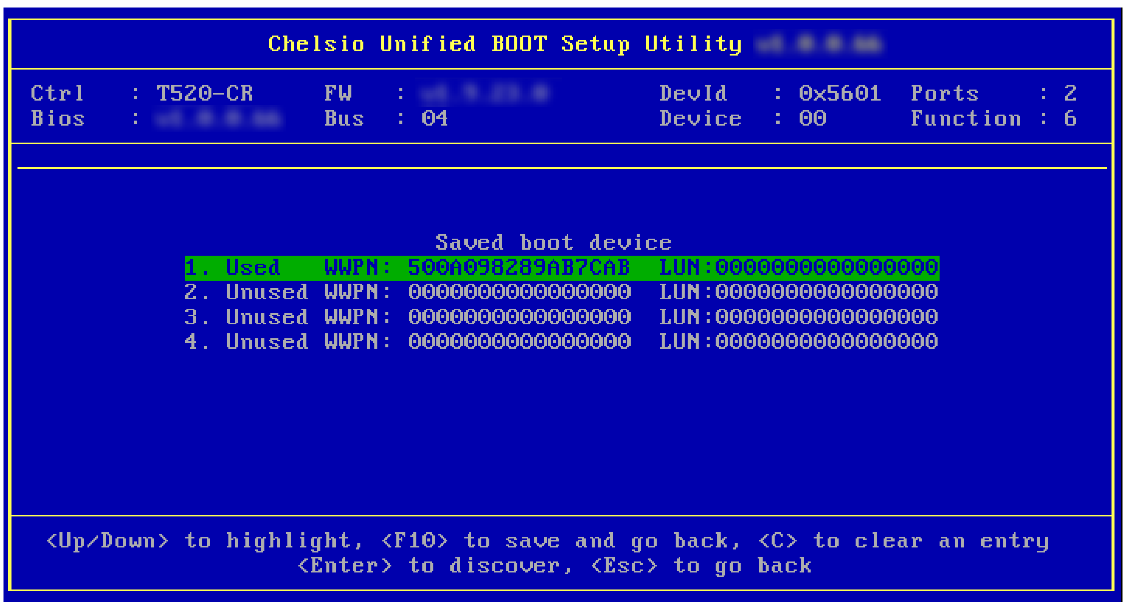


xiv. List of discovered targets will be displayed. Highlight a target using the arrow keys and hit [Enter] to select.

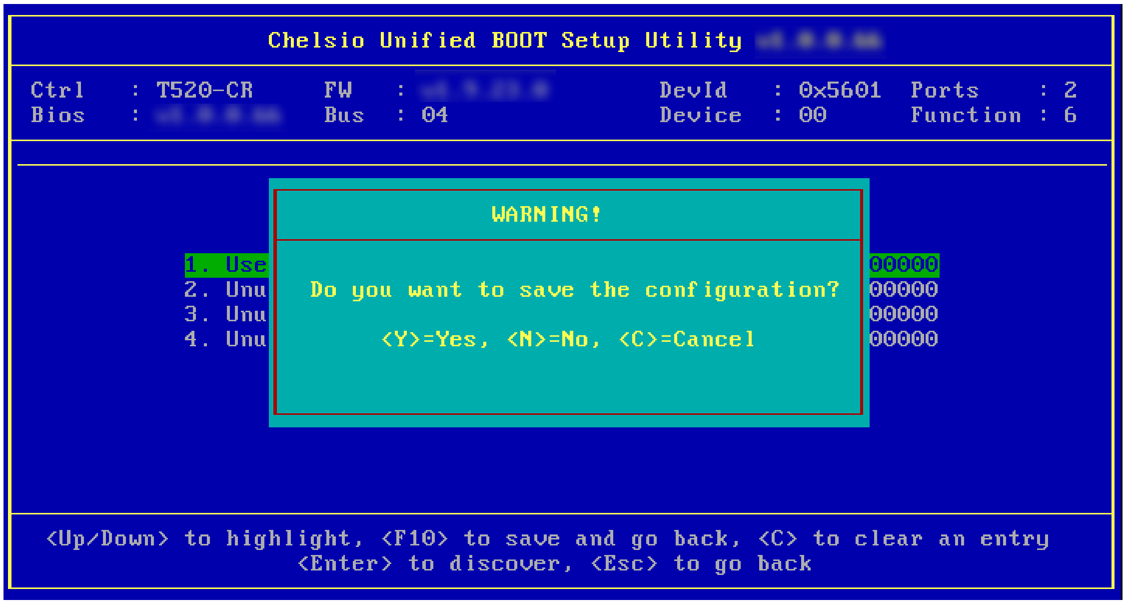


xv. From the list of LUNs displayed for the selected target, choose one on which operating system has to be installed. Hit [Enter].



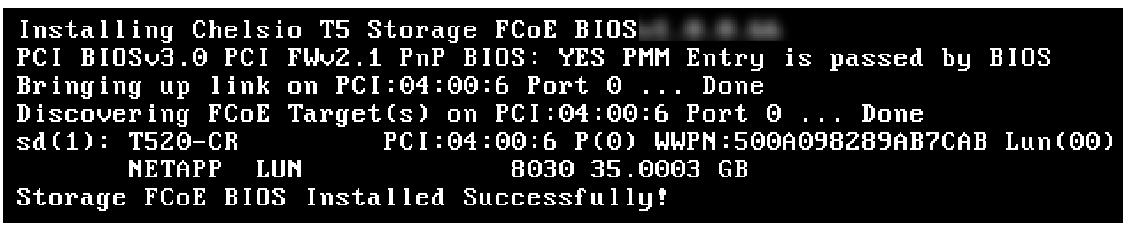


xvi. Hit [F10] or [Esc] and then [Y] to save the configuration.

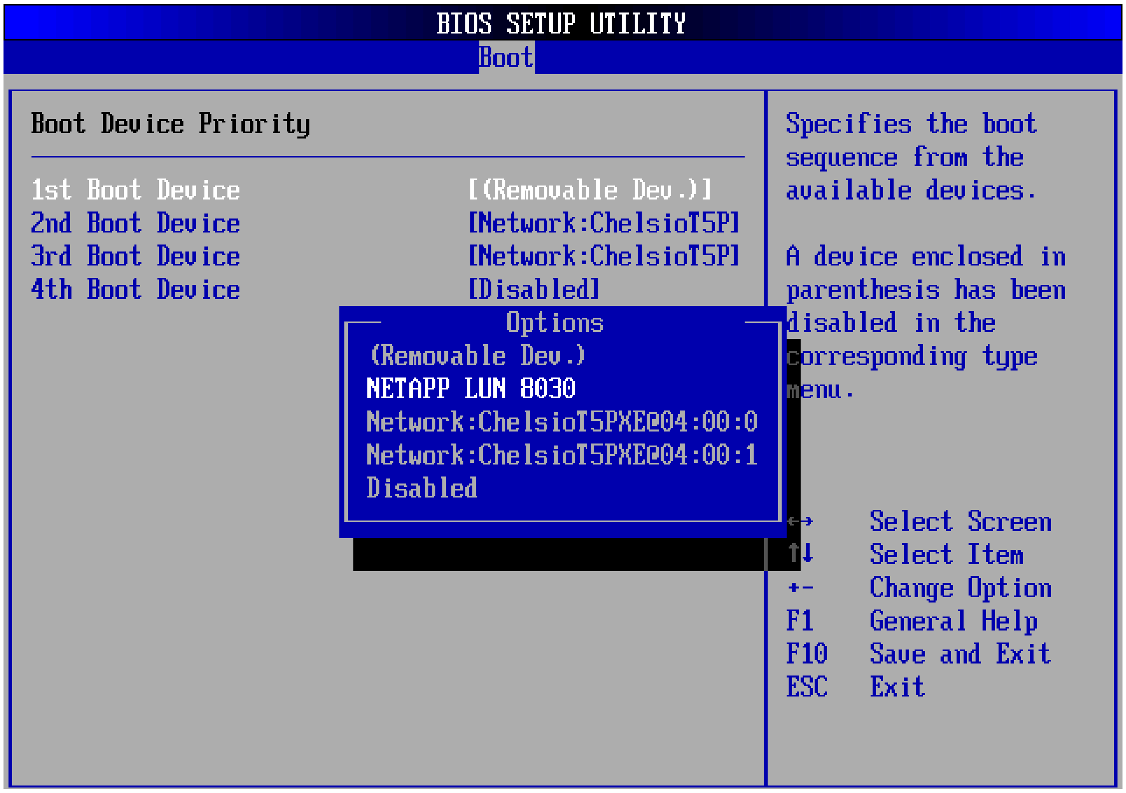


xvii. Reboot the machine.

xviii. During POST, allow the Chelsio option ROM to discover FCoE targets.



xix. Enter BIOS setup and choose FCoE disk discovered via Chelsio adapter as the first boot device.



xx. Reboot and boot from the FCoE disk or install the required OS using PXE.

5.2. uEFI FCoE Boot

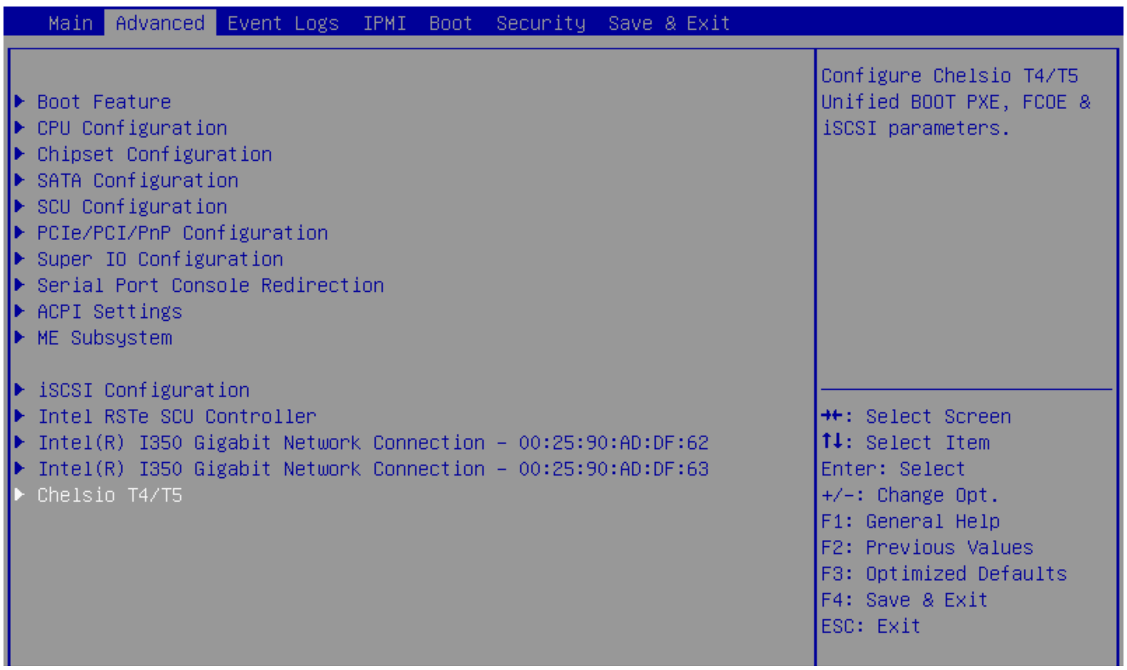
Important

- *Only uEFI v2.1 and v2.3.1 supported.*
- *Any other uEFI version is NOT SUPPORTED and may render your system unusable.*

i. Reboot the system and go into BIOS setup.

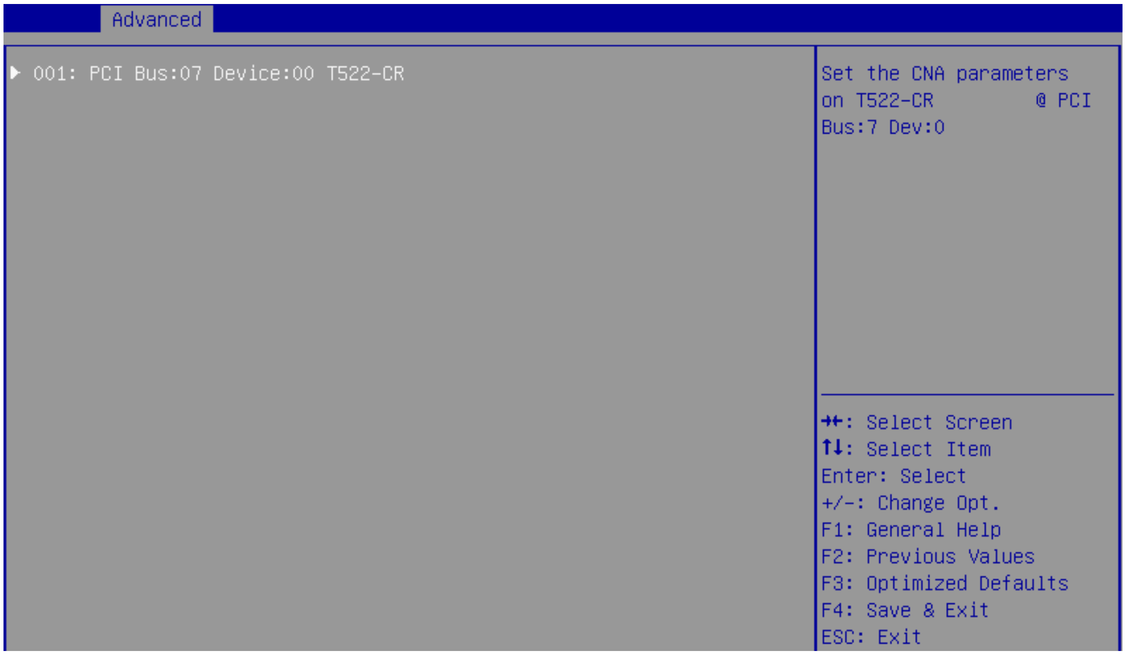
ii. Disable Secure Boot, if not already done.

iii. Select **Chelsio T4/T5** and press [Enter]

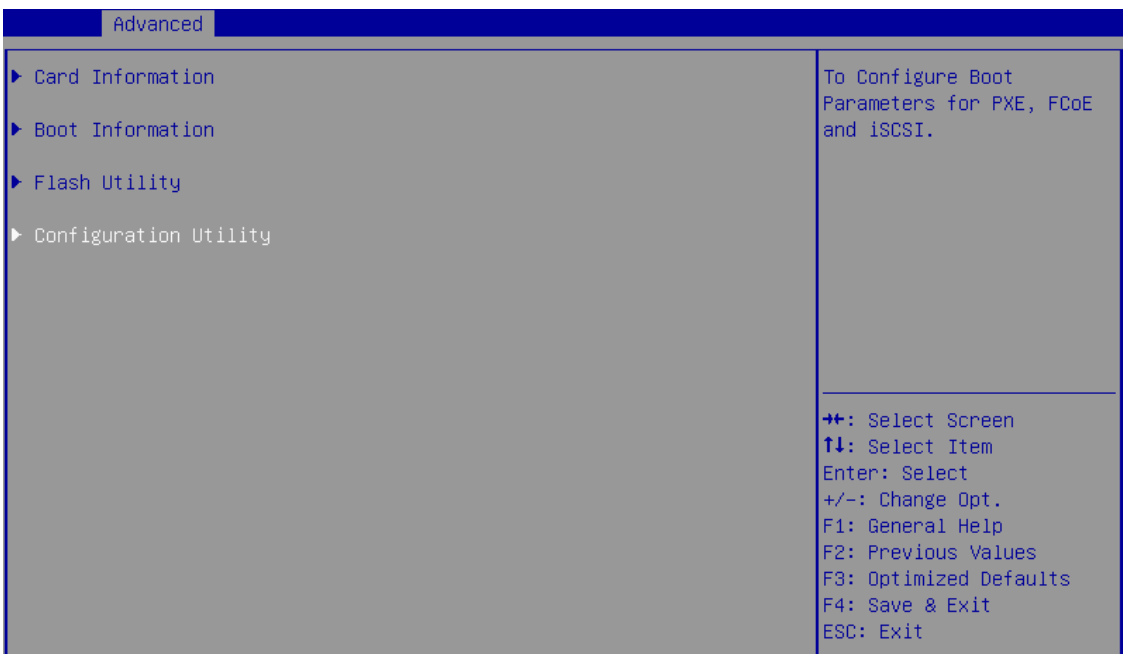


Note *If Chelsio T4/T5 is not listed, please ensure that Chelsio uEFI driver is loaded correctly as mentioned* ***here*** *in the* ***Flashing Firmware and Option ROM*** *section.*

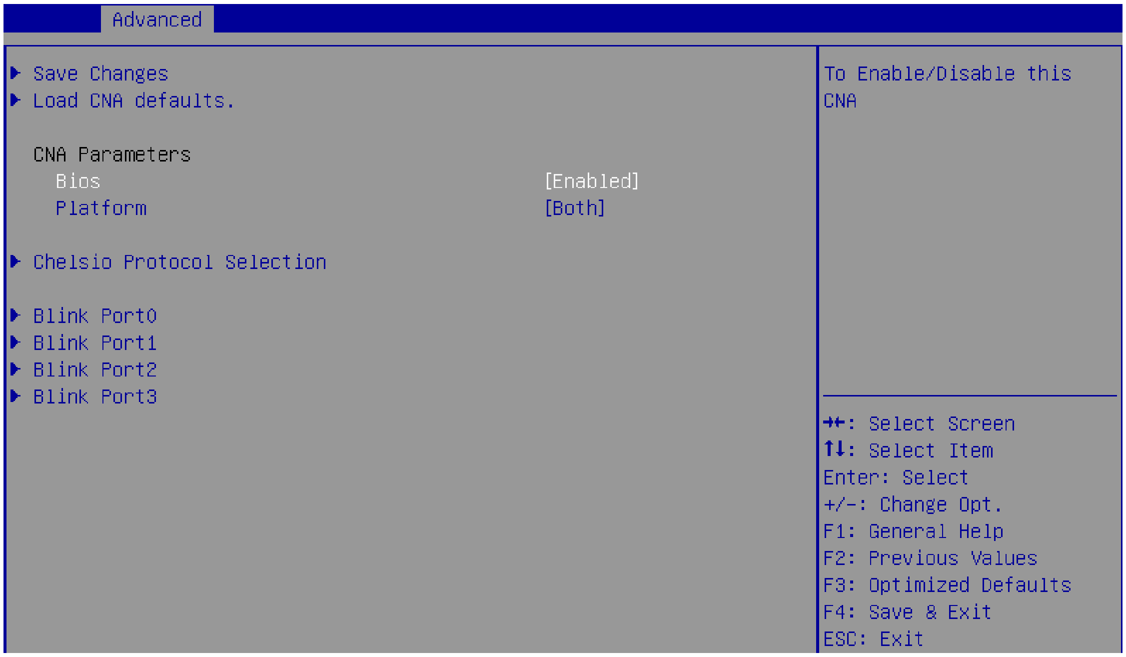
iv. Select the Chelsio adapter to be configured and press [Enter].



v. Select **Configuration Utility** and press [Enter].

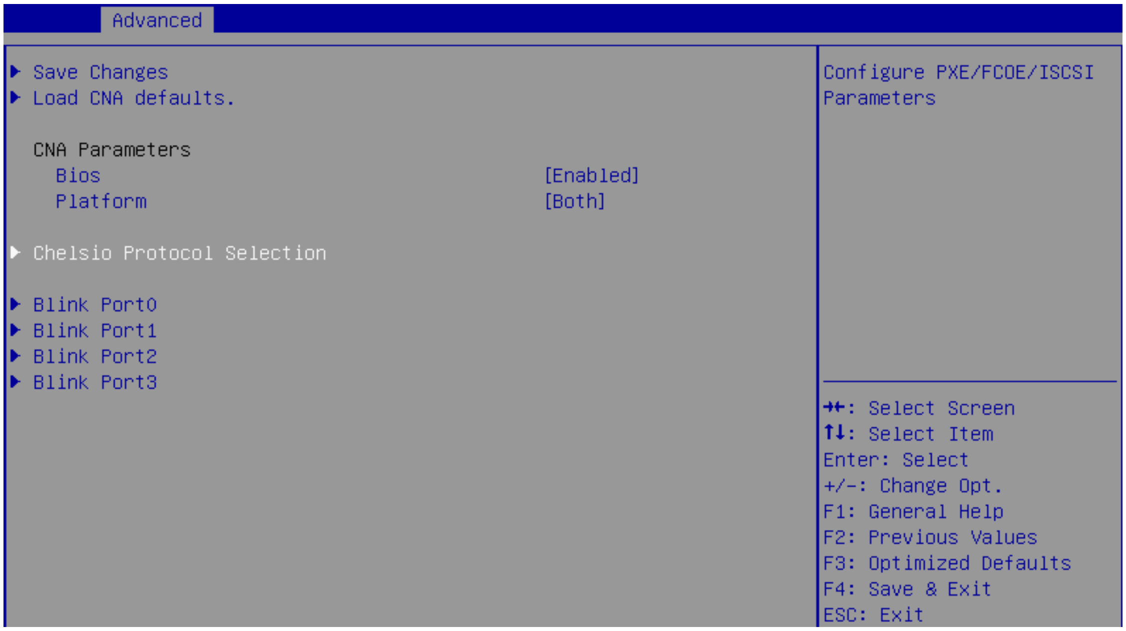


vi. Enable adapter BIOS if not already enabled.

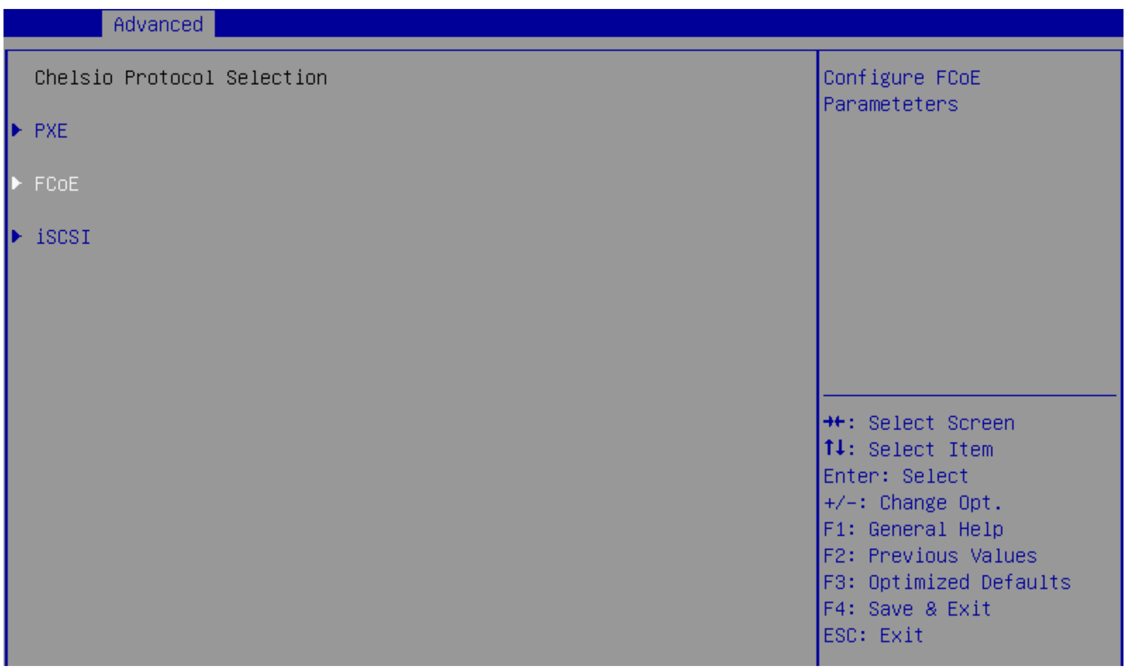


Note *It is highly recommended that you use the* ***Save Changes*** *option every time a parameter/option is changed.*

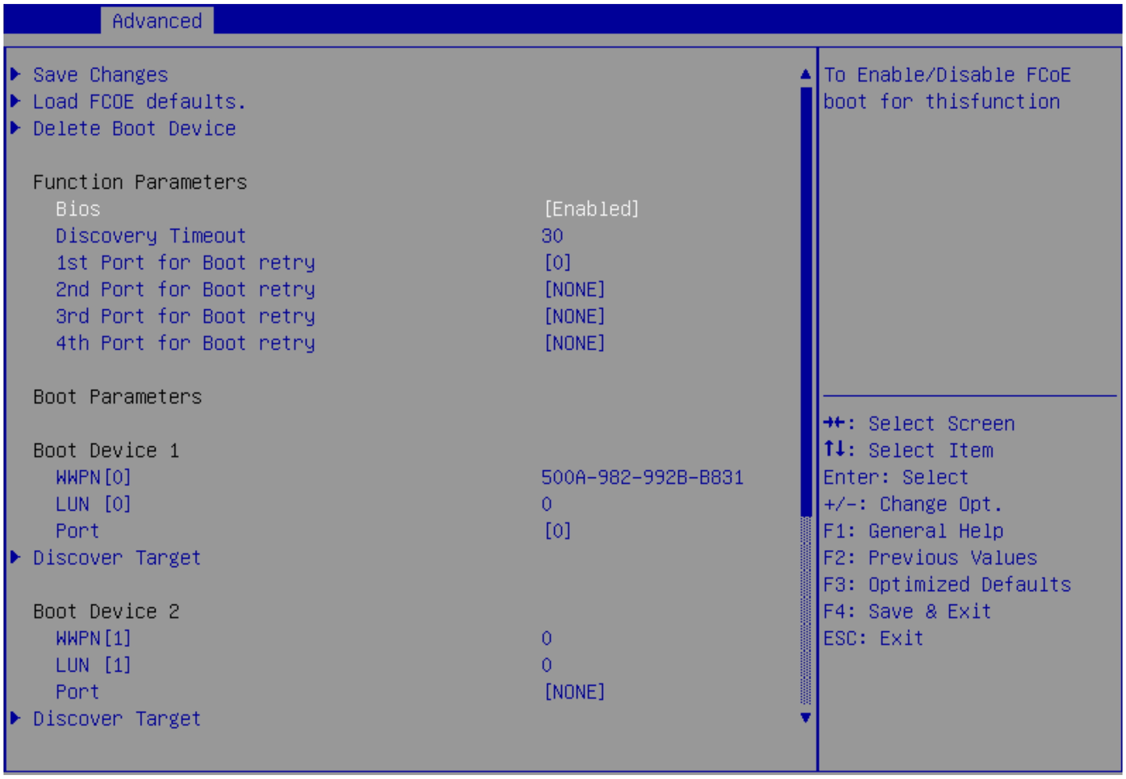
vii. Select **Chelsio Protocol Selection** and press [Enter].



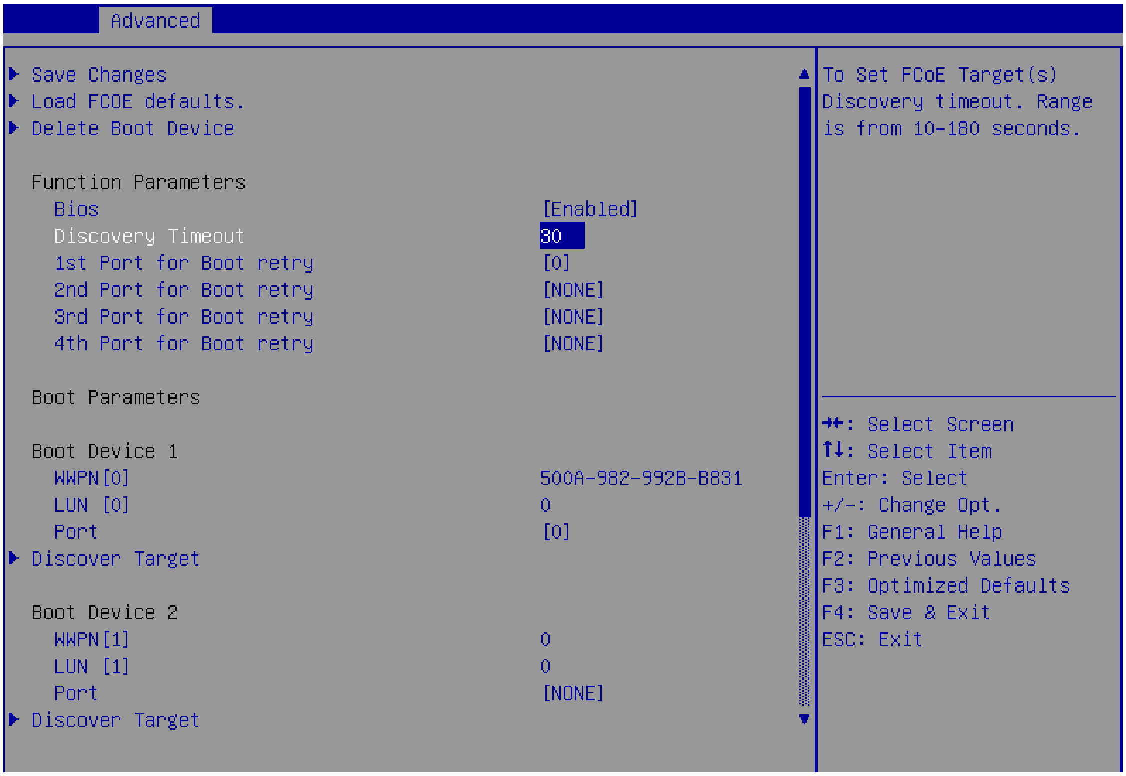
viii. Select **FCoE** and press [Enter].



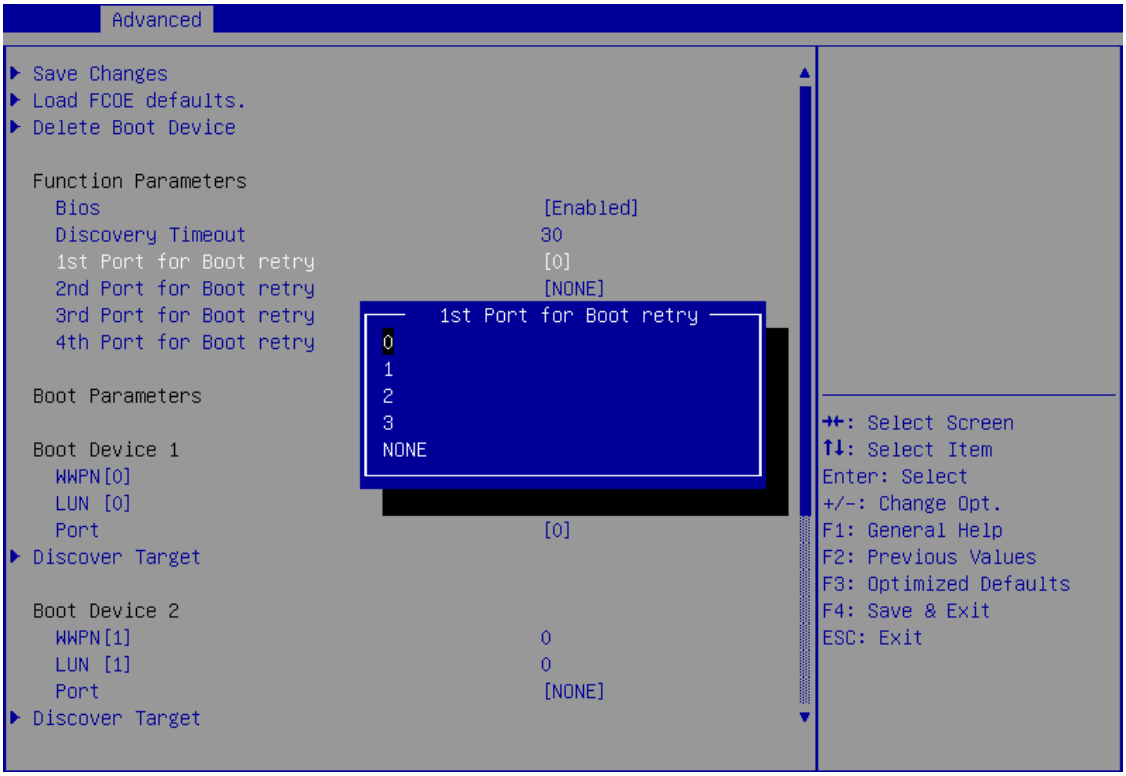
ix. Under **Function Parameters**, enable FCoE BIOS, if not already enabled.



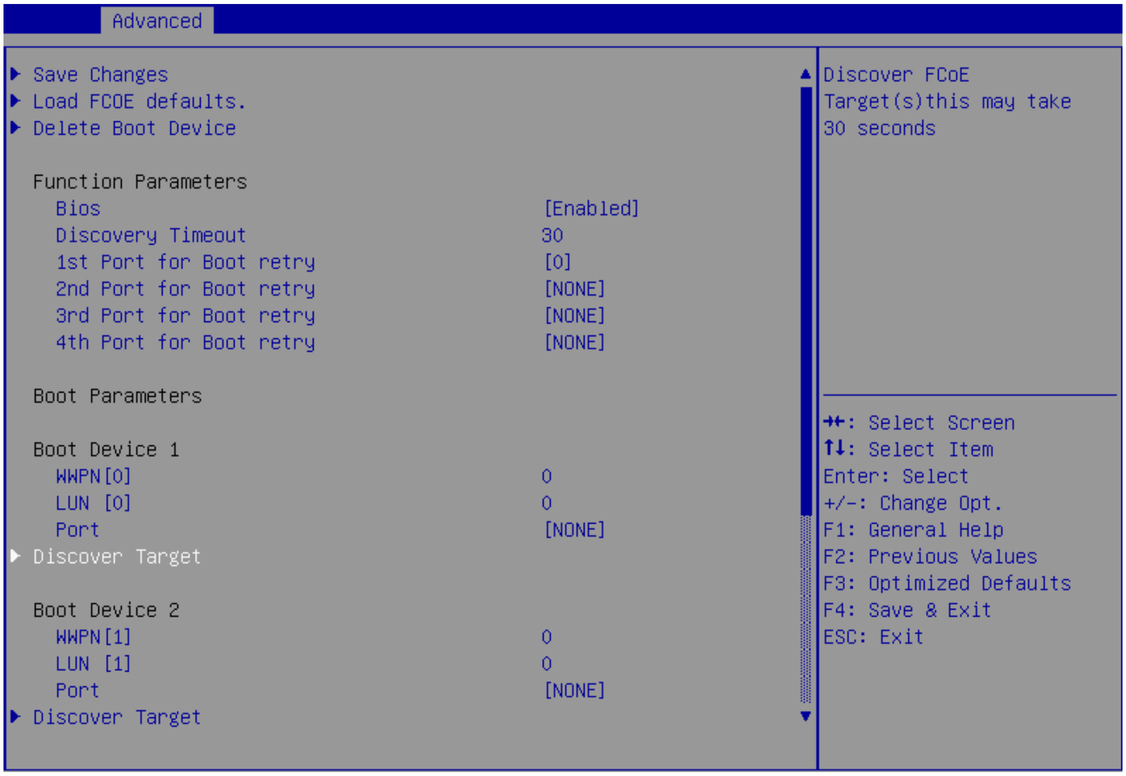
x. Set discovery timeout to a suitable value. Recommended value is >= 30



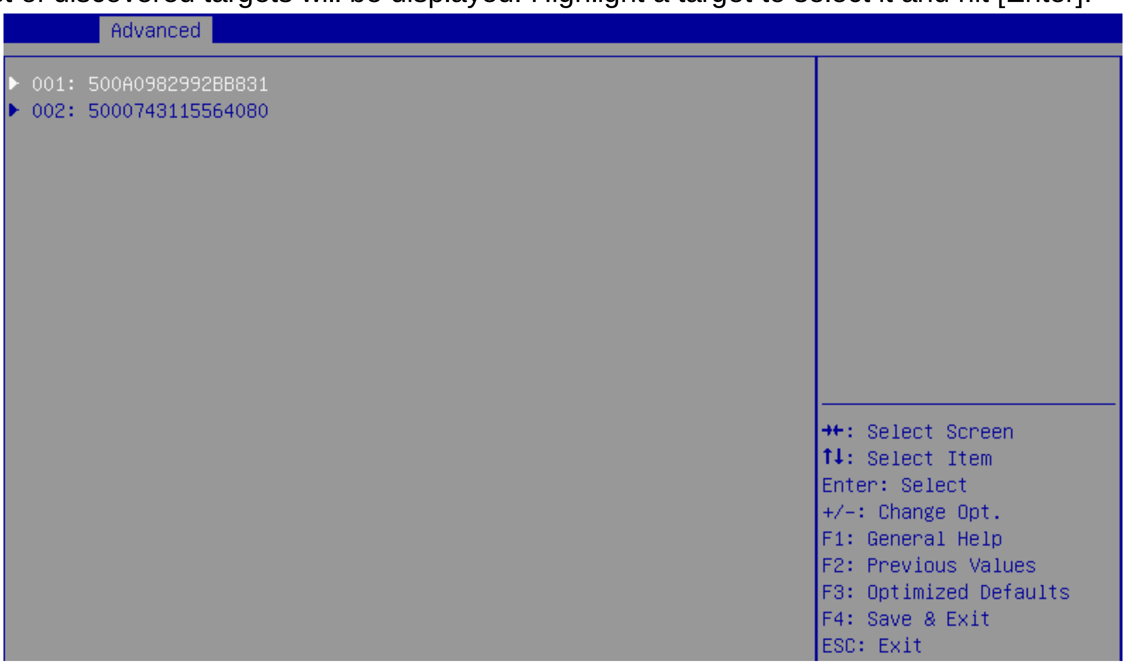
xi. Choose the order of the ports to discover FCoE targets.



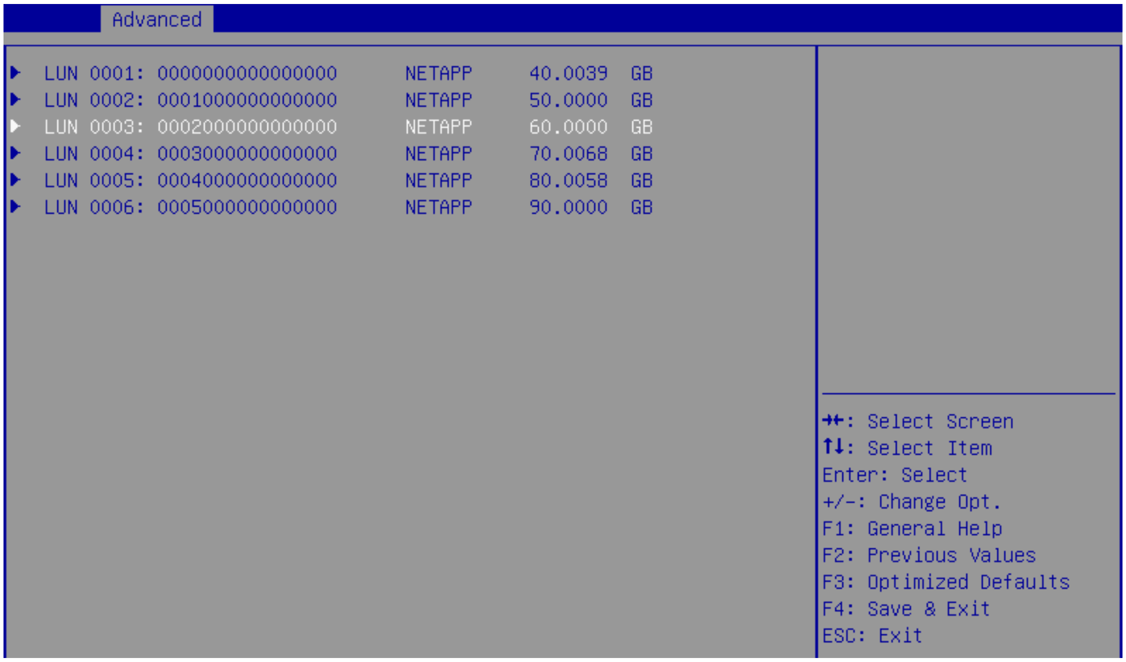
xii. Under the first boot device, select **Discover Target** and press [Enter] to discover FC/FCoE targets connected to the switch. Wait till all reachable targets are discovered.



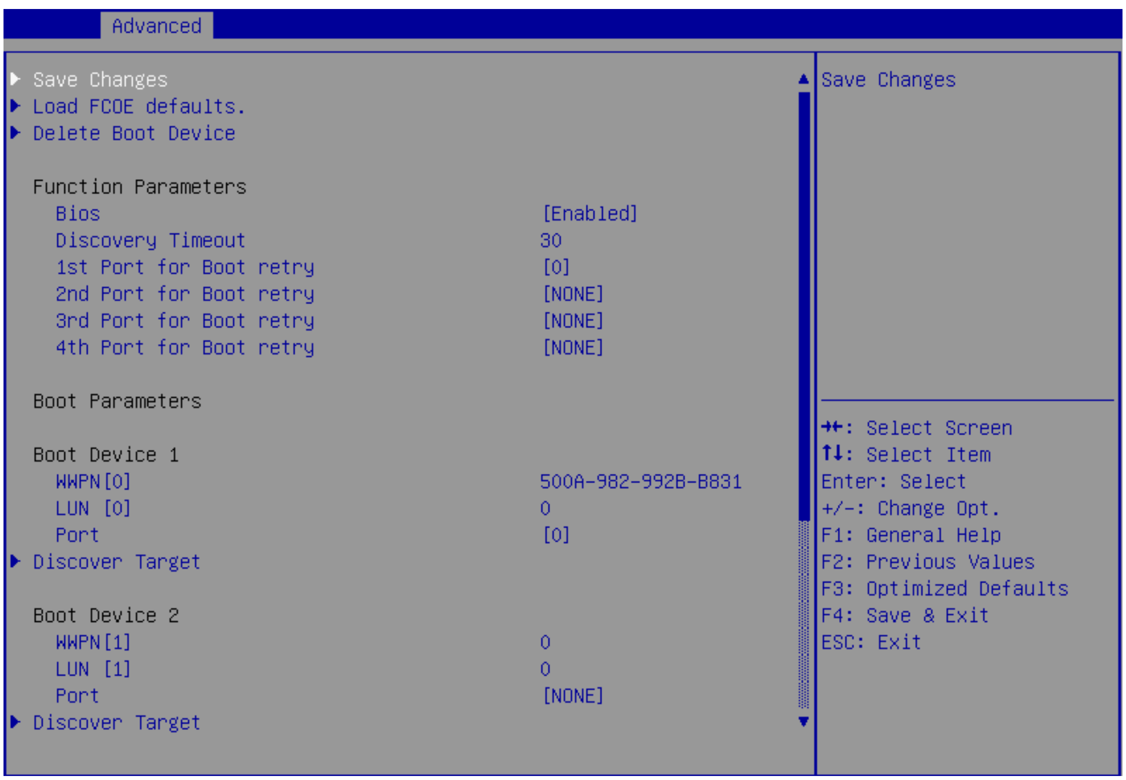
xiii. List of discovered targets will be displayed. Highlight a target to select it and hit [Enter].



xiv. List of LUNs for the selected target will be displayed. Highlight a LUN to select it and hit [Enter].

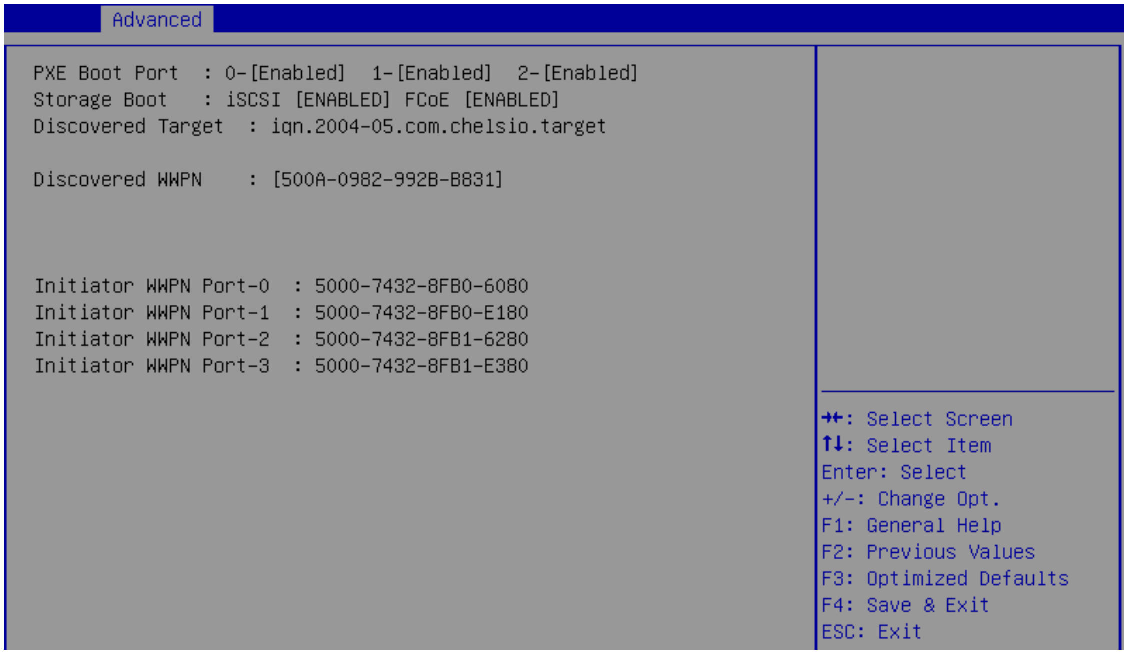


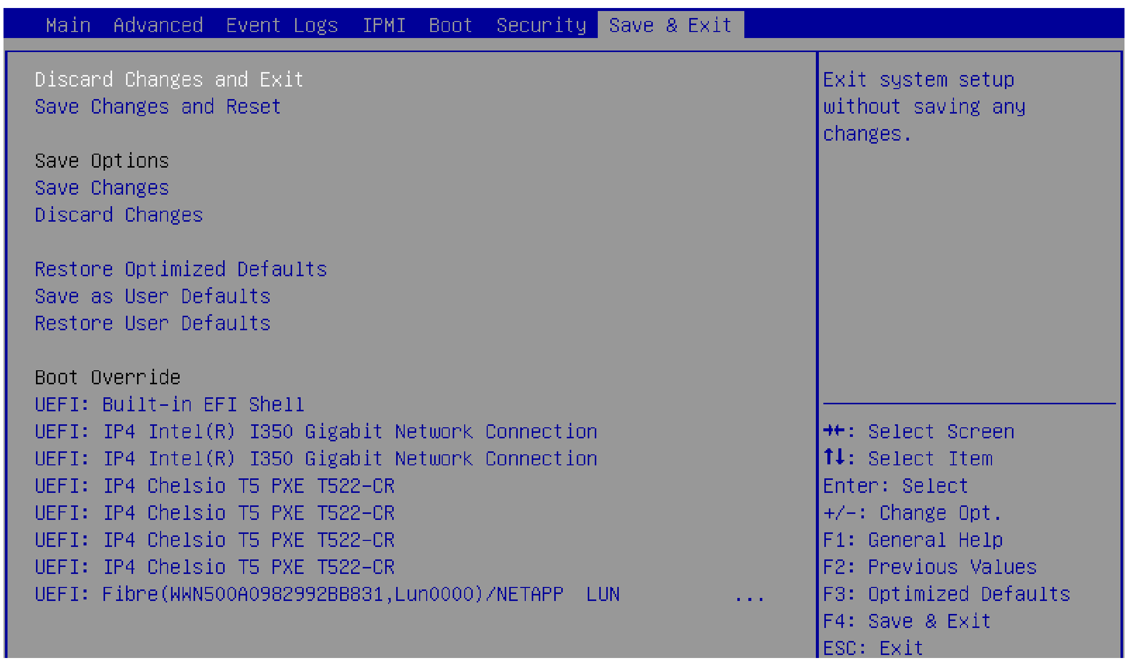
xv. Select **Save Changes** and press [Enter].



xvi. Reboot the system for changes to take effect.

xvii. The discovered LUN should appear in the **Boot Configuration** section and system BIOS section.





xviii. Select the LUN as the first boot device and exit from BIOS.

xix. Either boot from the LUN or install the required OS.

6. iSCSI boot process

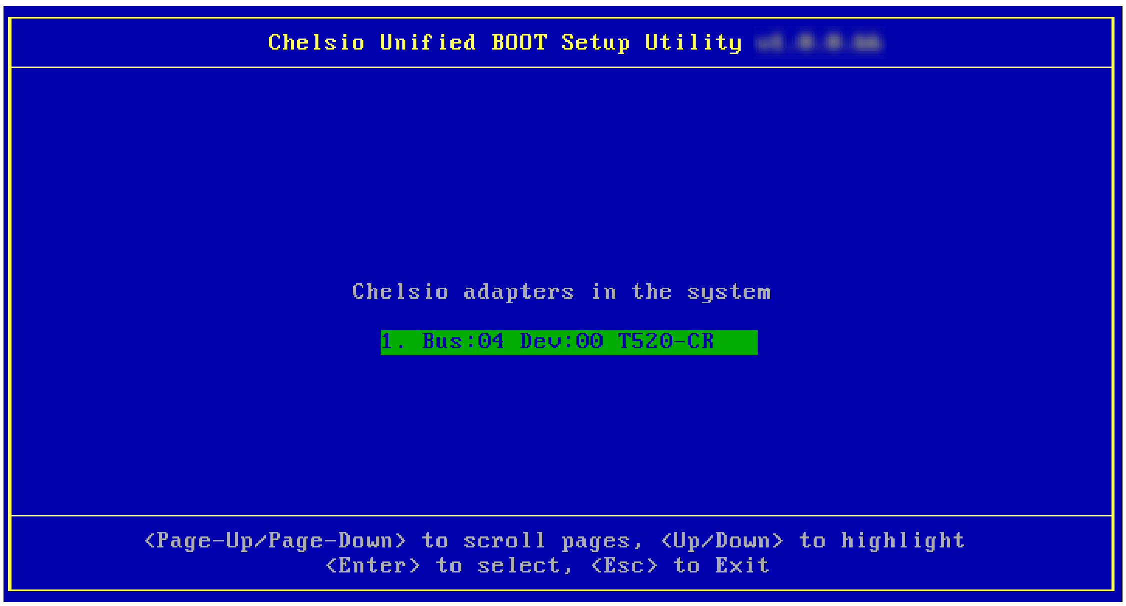
Before proceeding, please ensure that the Chelsio CNA has been flashed with the provided firmware and option ROM (See **Flashing firmware and option ROM**).

6.1. Legacy iSCSI boot

i. Reboot the system.

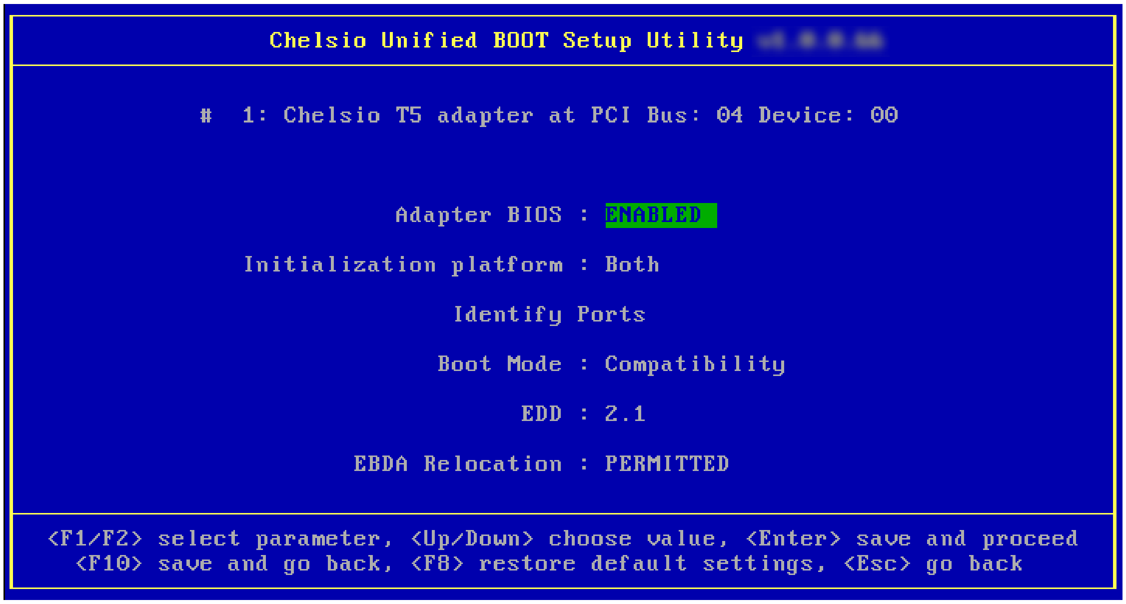
ii. Press [Alt+C] when the message "*Chelsio Unified Boot BIOS vX.X.X.XX, Copyright (C) 2003-2016 Chelsio Communications Press <Alt-C> to Configure T4/T5 Card(s)*. *Press <Alt-S> to skip B*IOS" appears on the screen to enter the configuration utility.

iii. The configuration utility will appear as below:



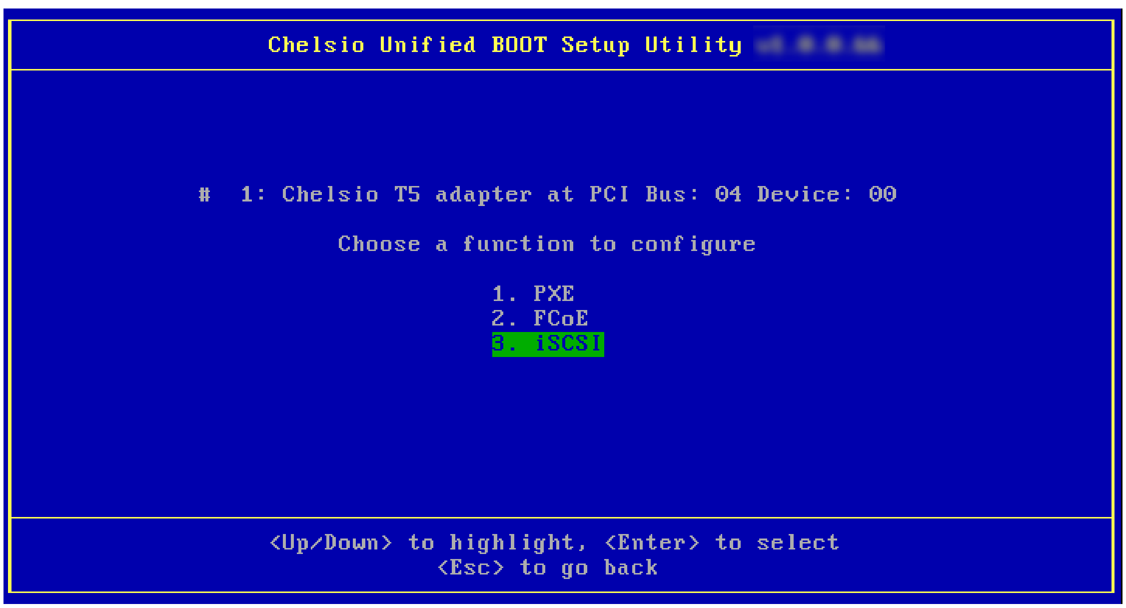
iv. Choose the CNA on which you flashed the option ROM image. Hit [Enter].

v. Enable the adapter BIOS if not already enabled. Hit [Enter].

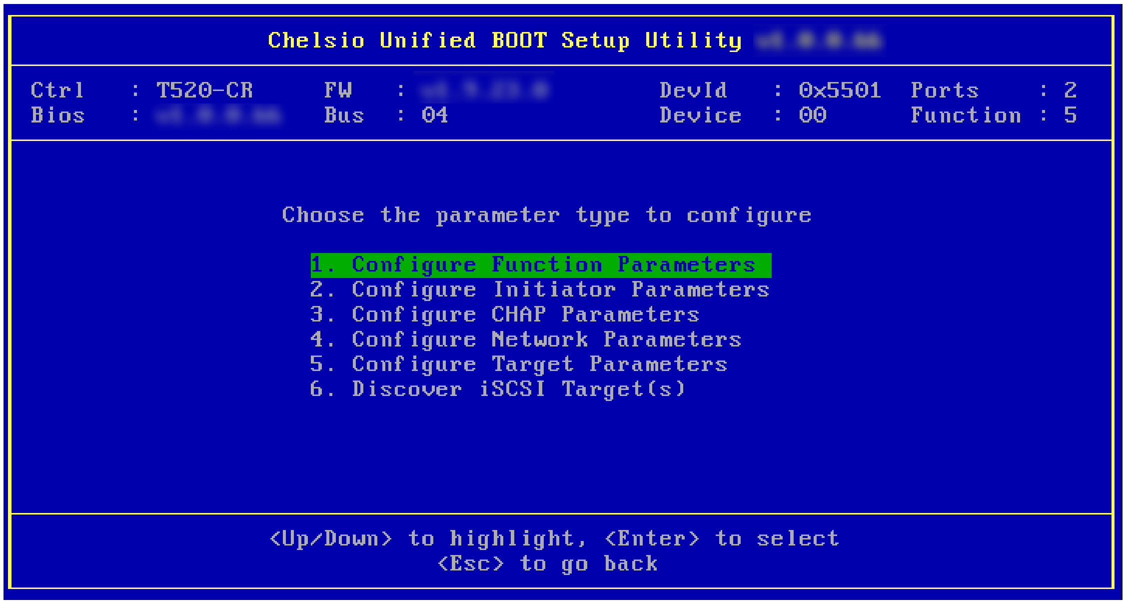


Note *Use the default values for Boot Mode, EDD and EBDA Relocation parameters, unless instructed otherwise.*

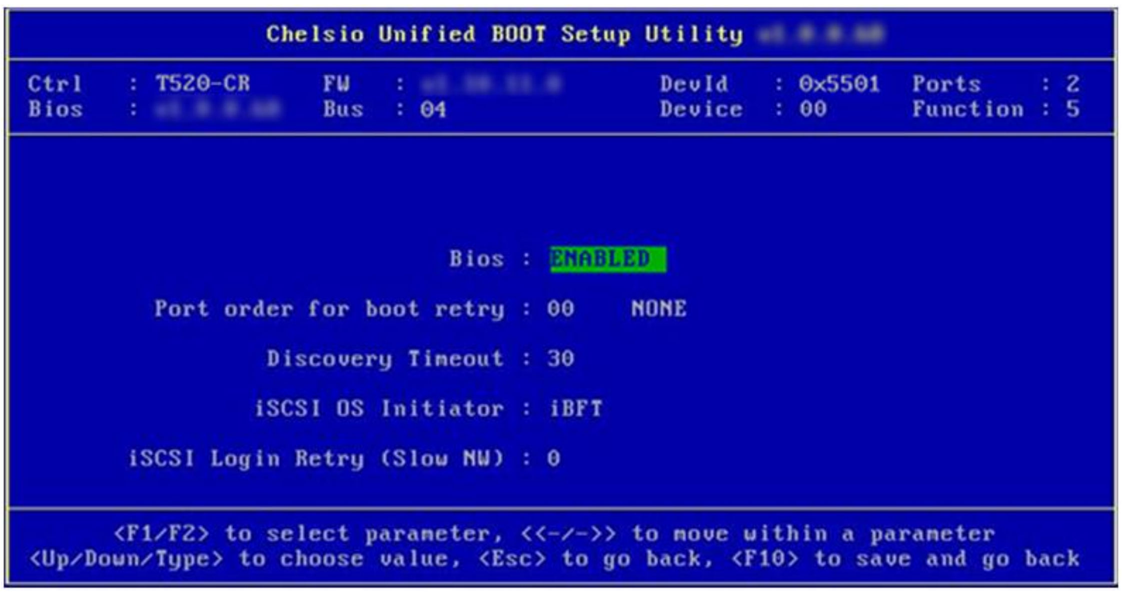
vi. Choose *iSCSI* from the list to configure and hit [Enter].



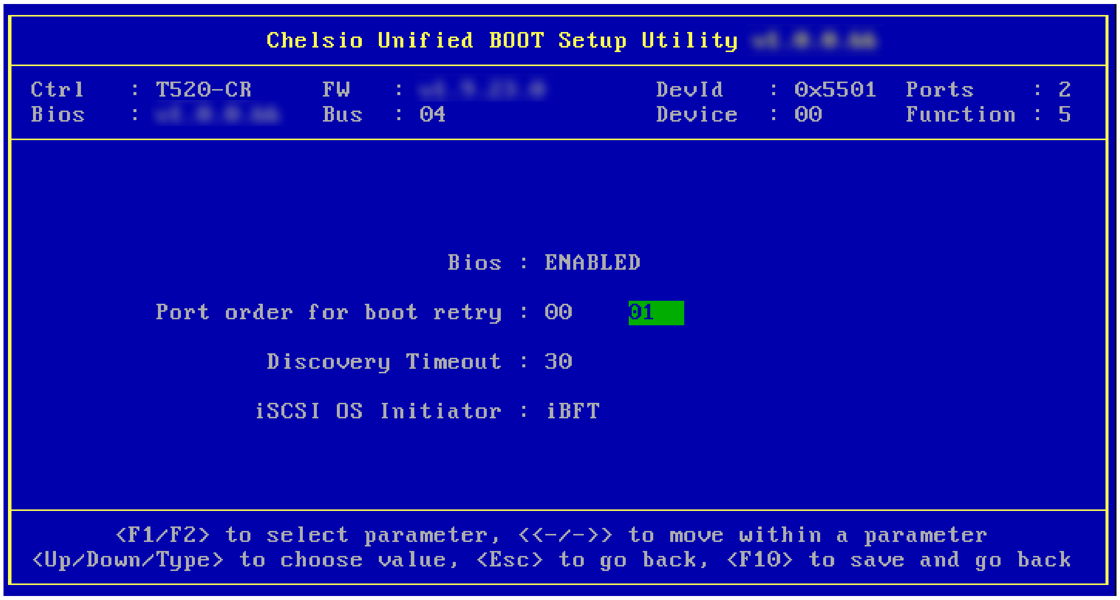
vii. Choose the first option, **Configure Function Parameters**, from the list of parameter type and hit [Enter].



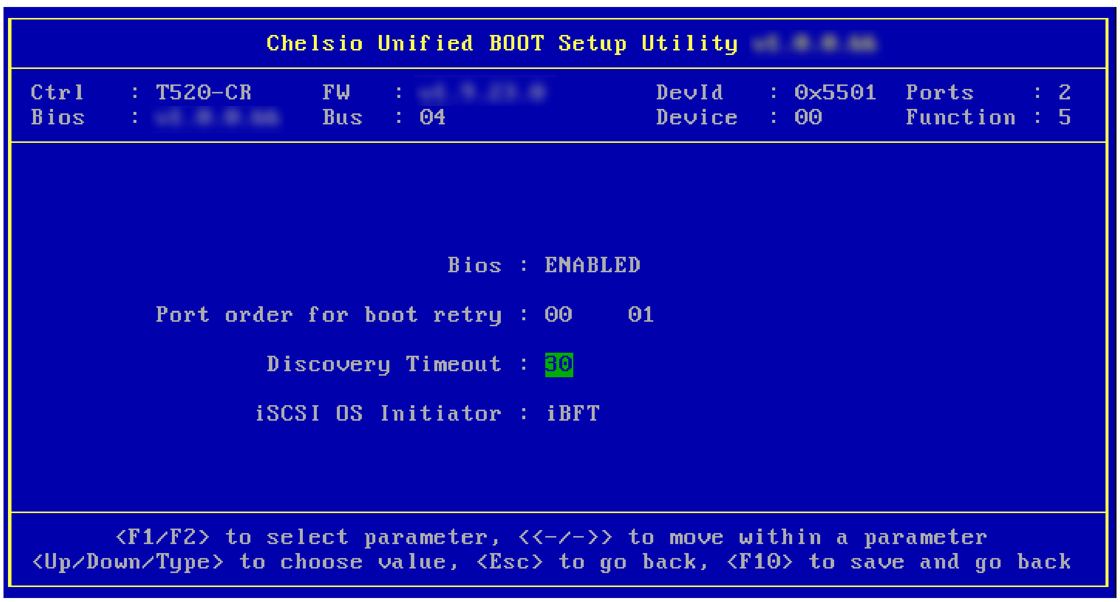
viii. Enable iSCSI BIOS if not already enabled. iBFT (iSCSI Boot Firmware Table) will be selected by default. You can also configure the number of iSCSI login attempts (retries) in case the network is unreachable or slow.



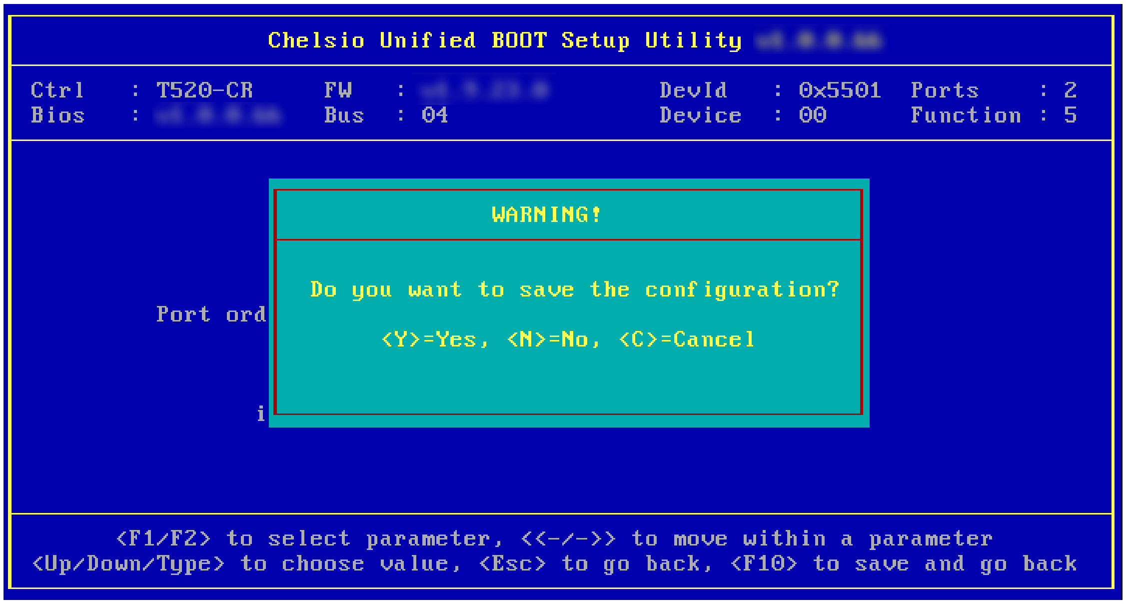
ix. Choose the order of the ports to discover iSCSI targets.



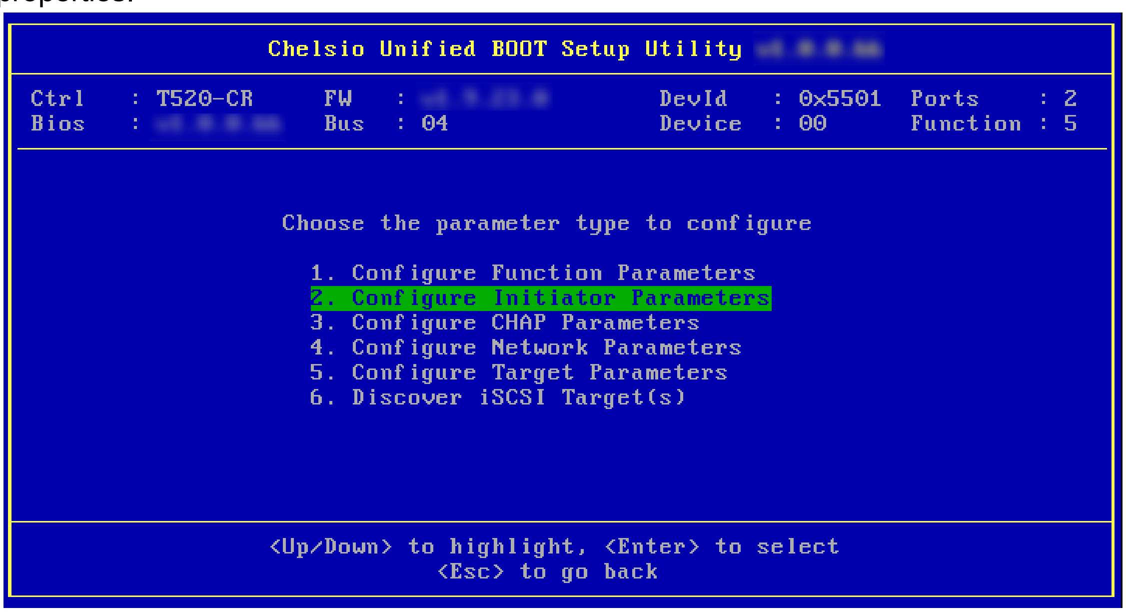
x. Set discovery timeout to a suitable value. Recommended value is >= 30.



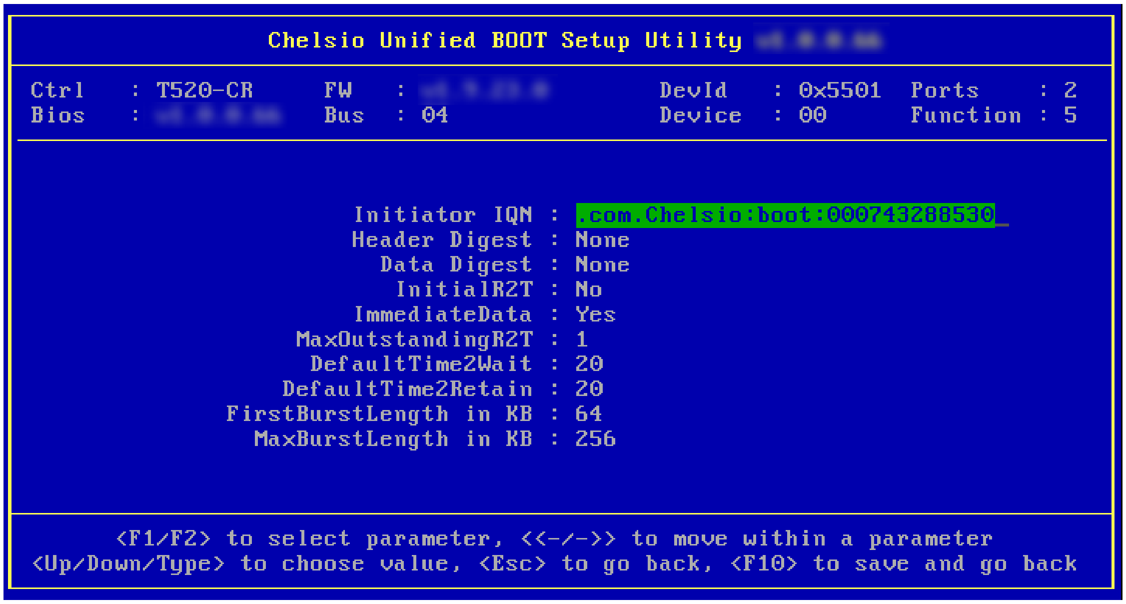
xi. Hit [Esc] and then [Y] to save the configuration.



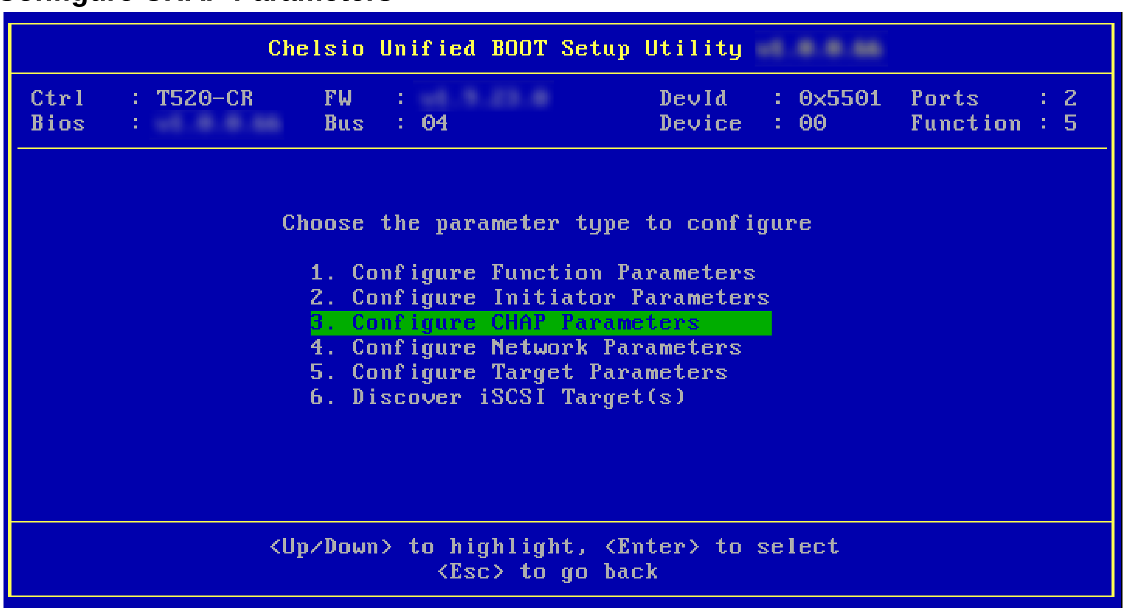
xii. Go back and choose **Configure Initiator Parameters** to configure initiator related properties.



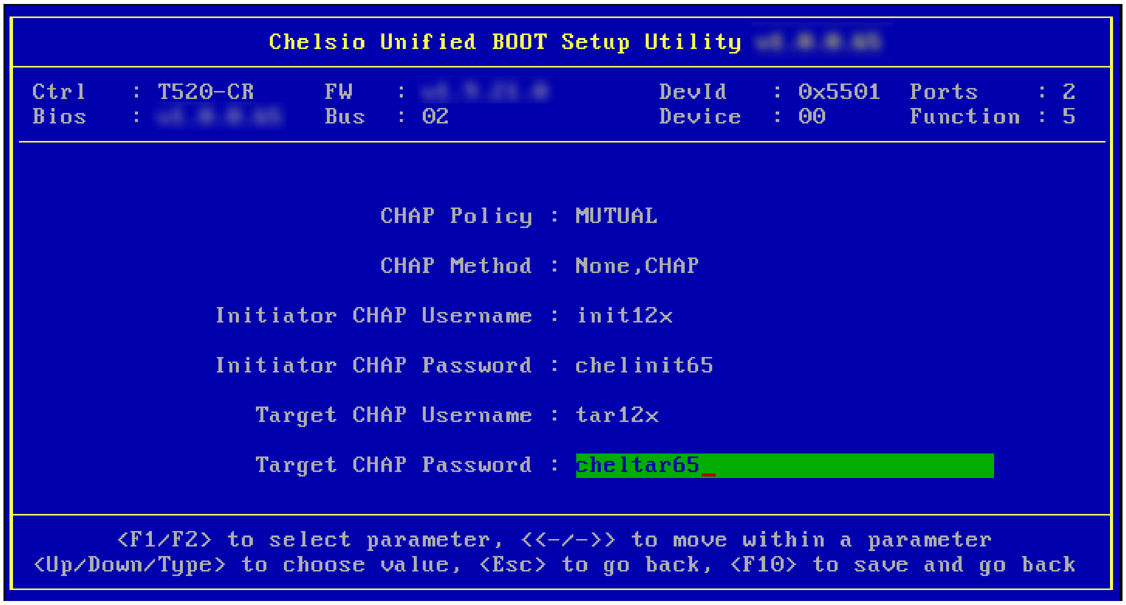
xiii. Initiator properties like IQN, Header Digest, Data Digest, etc. will be displayed. Change the values appropriately or continue with the default values. Hit [F10] to save.



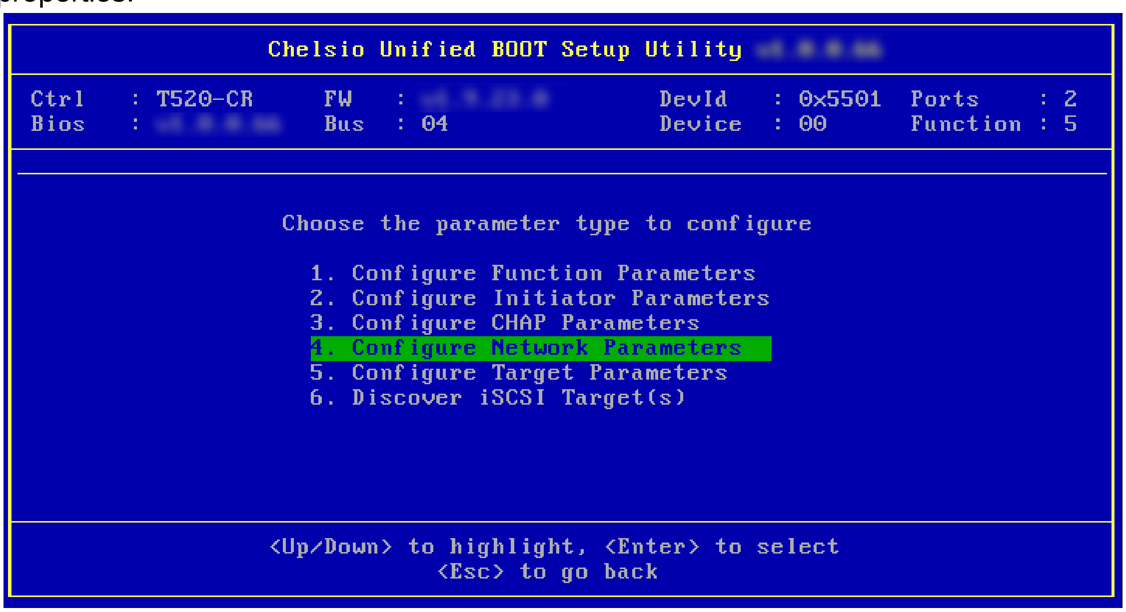
xiv. CHAP authentication is disabled by default. To enable and configure, go back and choose **Configure CHAP Parameters**



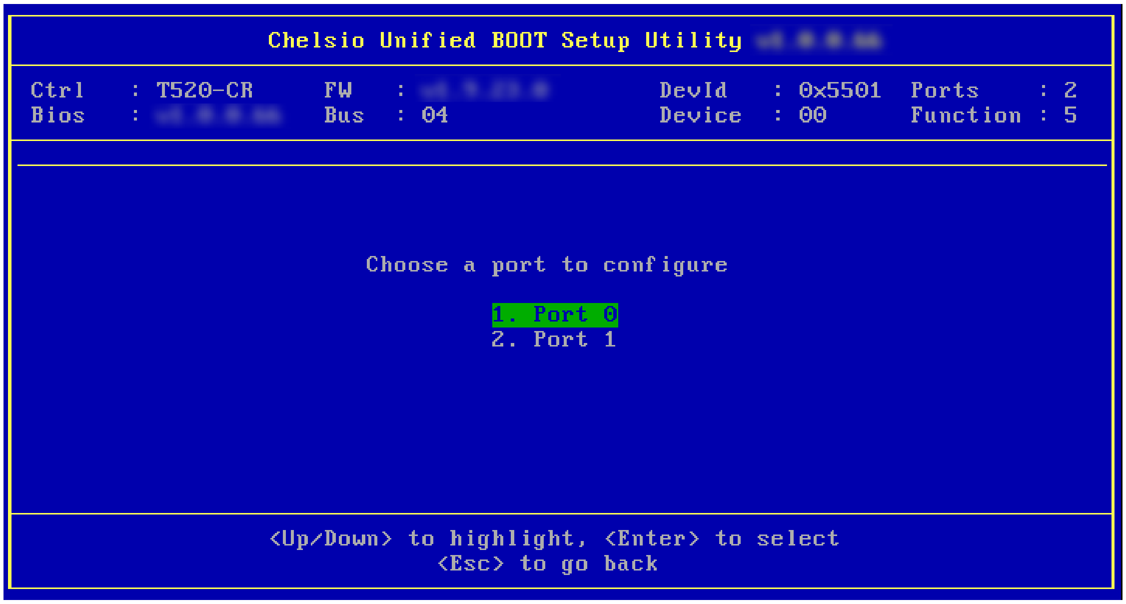
xv. Enable CHAP authentication by selecting *ENABLED* in the **CHAP Policy** field. Next, choose either *one-way* or *mutual* as the authentication method. Finally, provide Initiator and Target CHAP credentials according to the authentication method selected. Hit [F10] to save.



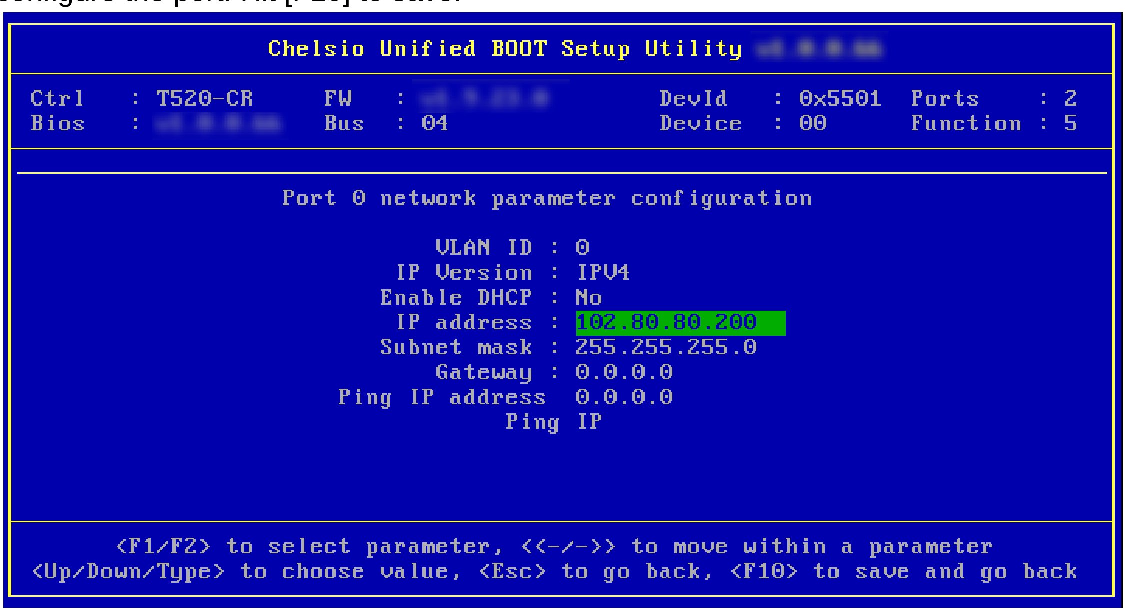
xvi. Go back and choose **Configure Network Parameters** to configure iSCSI Network related properties.



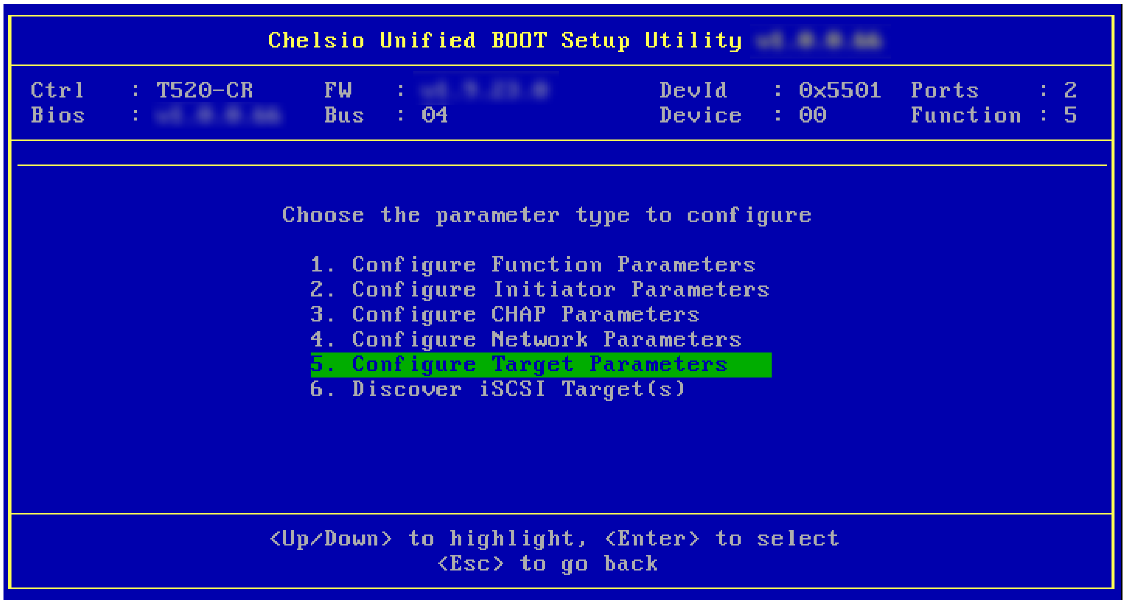
xvii. Select the port using which you want to connect to the target. Hit [Enter].



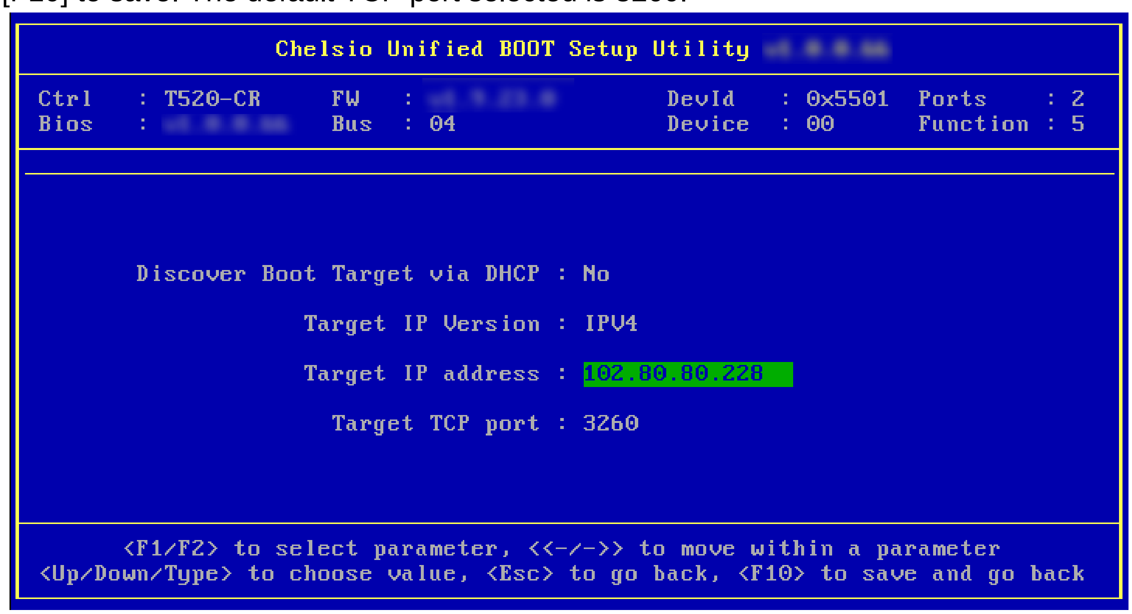
xviii. Select *Yes* in the **Enable DHCP** field to configure port using DHCP or *No* to manually configure the port. Hit [F10] to save.



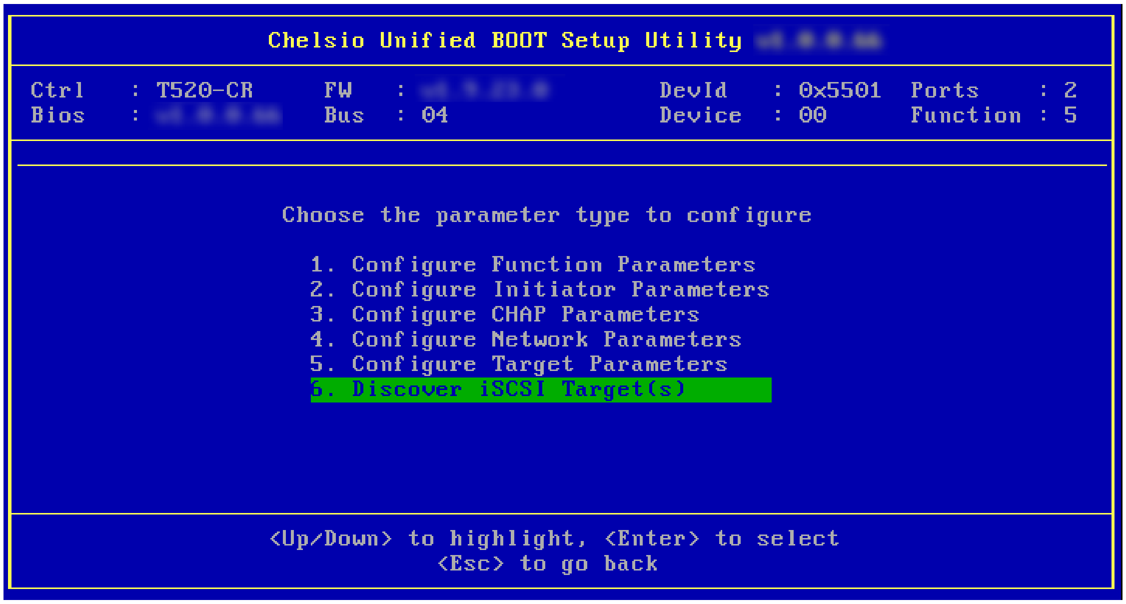
xix. Go back and choose **Configure Target Parameters** to configure iSCSI target related properties.



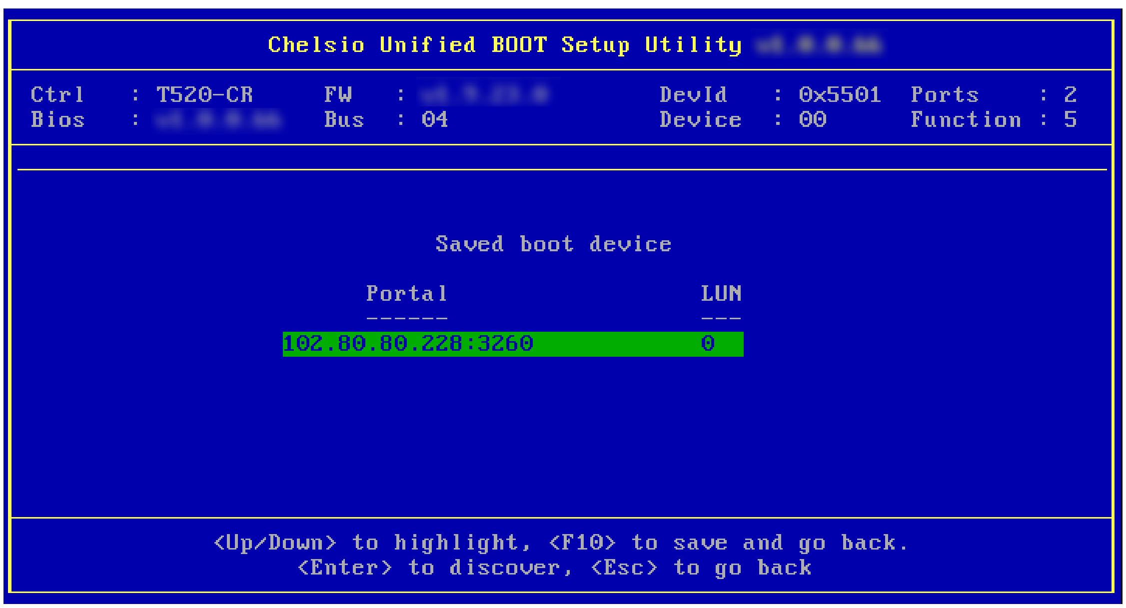
xx. If you want to discover target using DHCP, select *Yes* in the **Discover Boot Target via DHCP** field. To discover target via static IP, select *No* and provide the target IP and Hit [F10] to save. The default TCP port selected is 3260.



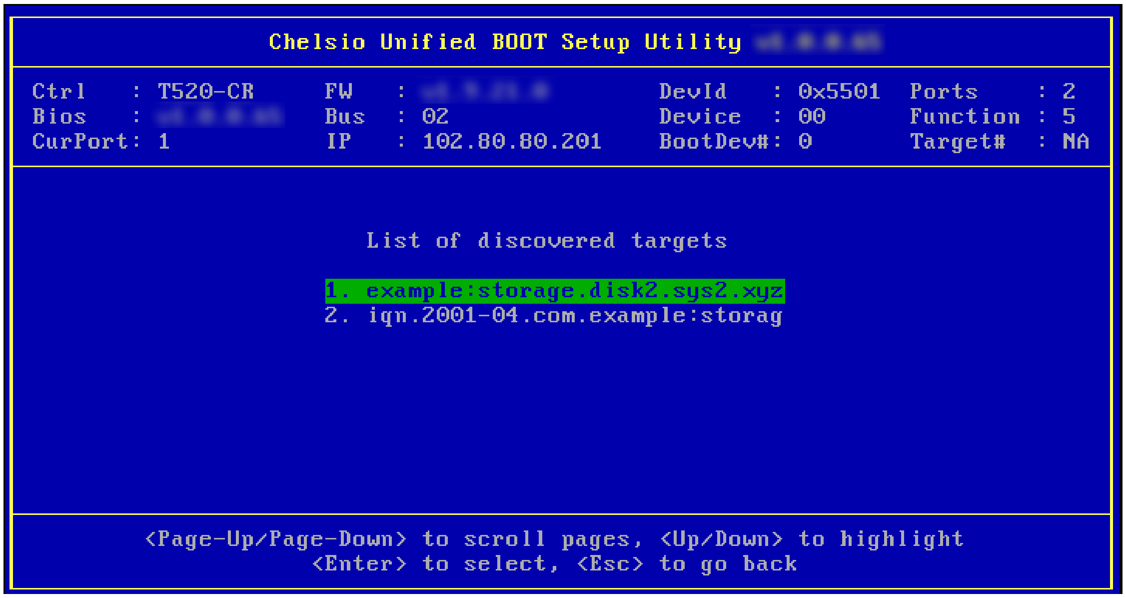
xxi. Go back and choose **Discover iSCSI Target (s)** to connect to a target.



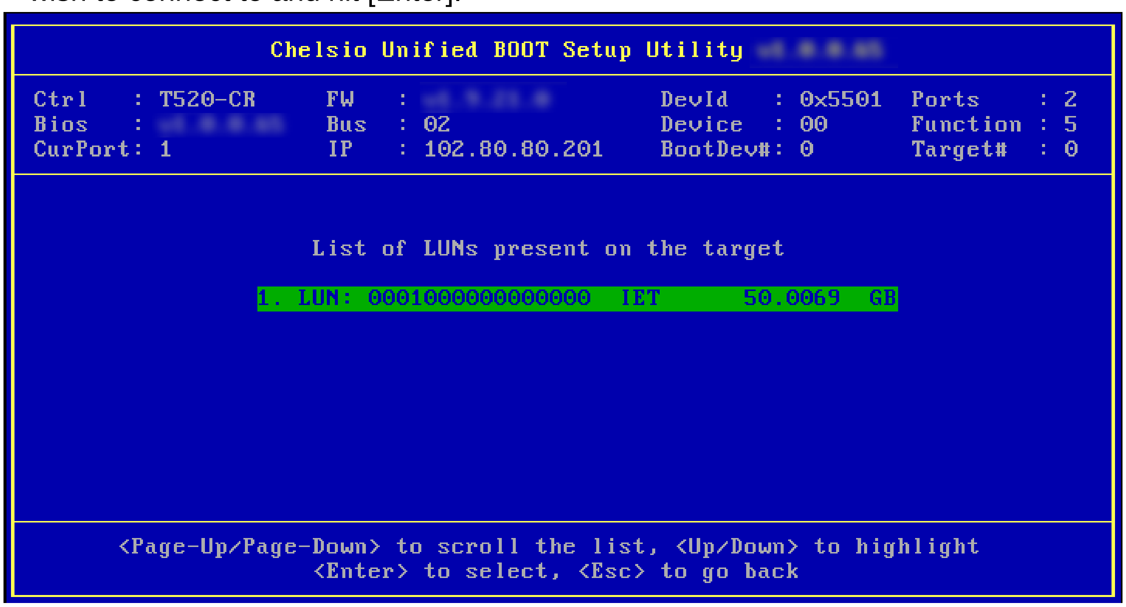
xxii. Select the portal group on which iSCSI service is provided by the target.



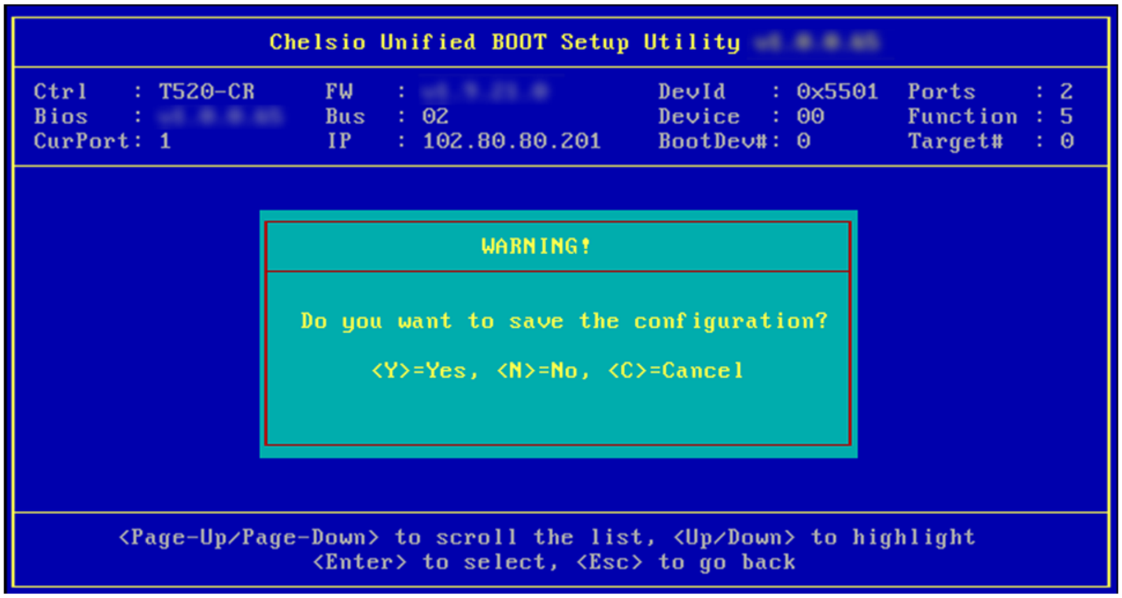
xxiii. A list of available targets will be displayed. Select the target you wish to connect to and hit [Enter].



xxiv. A list of LUNs configured on the selected target will be displayed. Select the LUN you wish to connect to and hit [Enter].

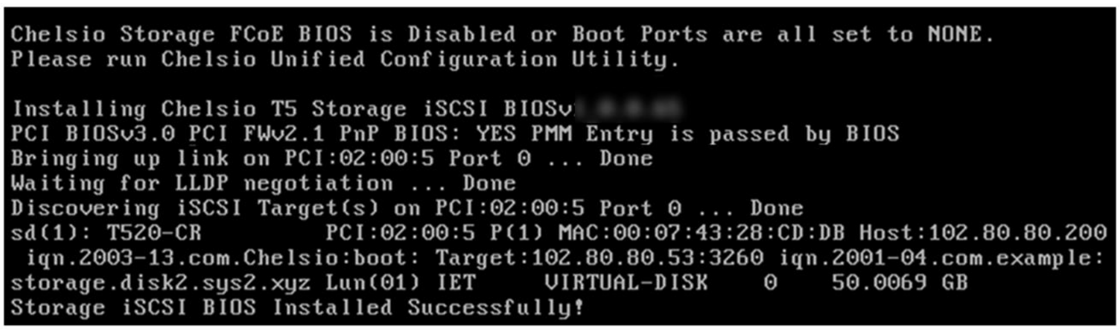


xxv. Hit [Esc] and then [Y] to save the configuration.

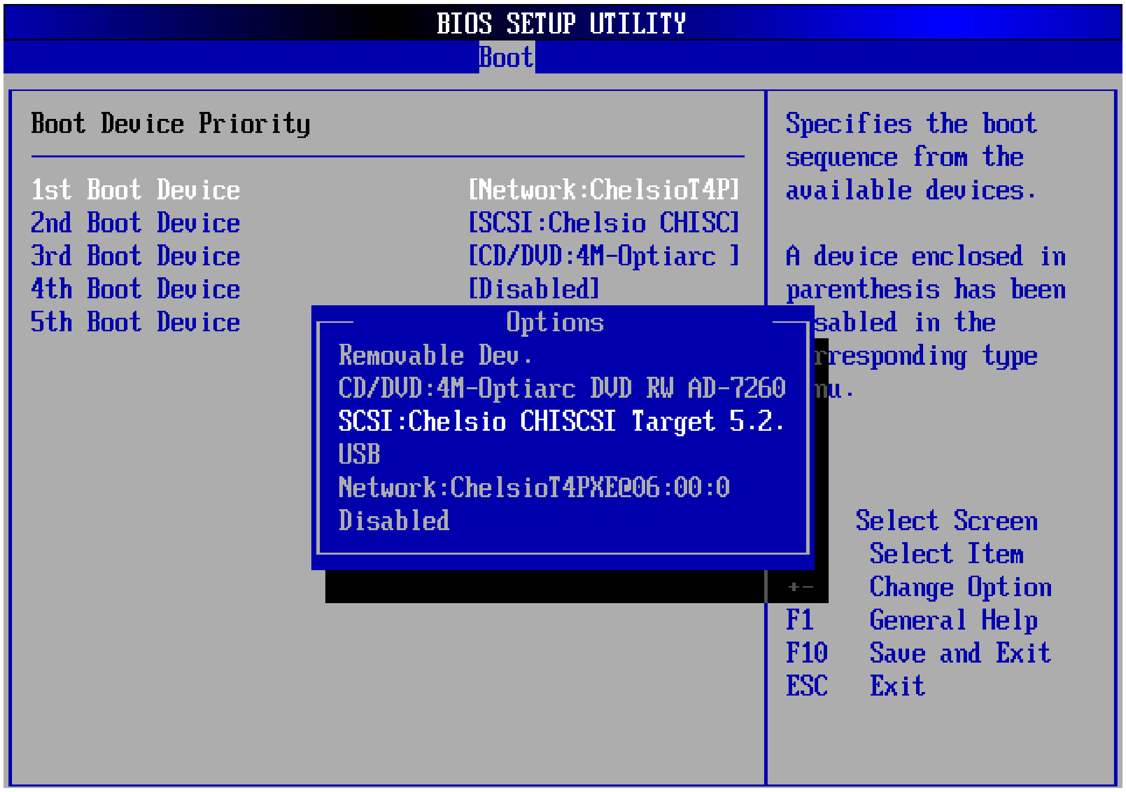


xxvi. Reboot the machine.

xxvii. During POST, allow the Chelsio option ROM to discover iSCSI targets.



xxviii. Enter BIOS setup and choose iSCSI target LUN discovered via Chelsio adapter as the first boot device.



xxix. Reboot and boot from the iSCSI Target LUN or install the required OS using PXE.

6.2. uEFI iSCSI Boot

Important

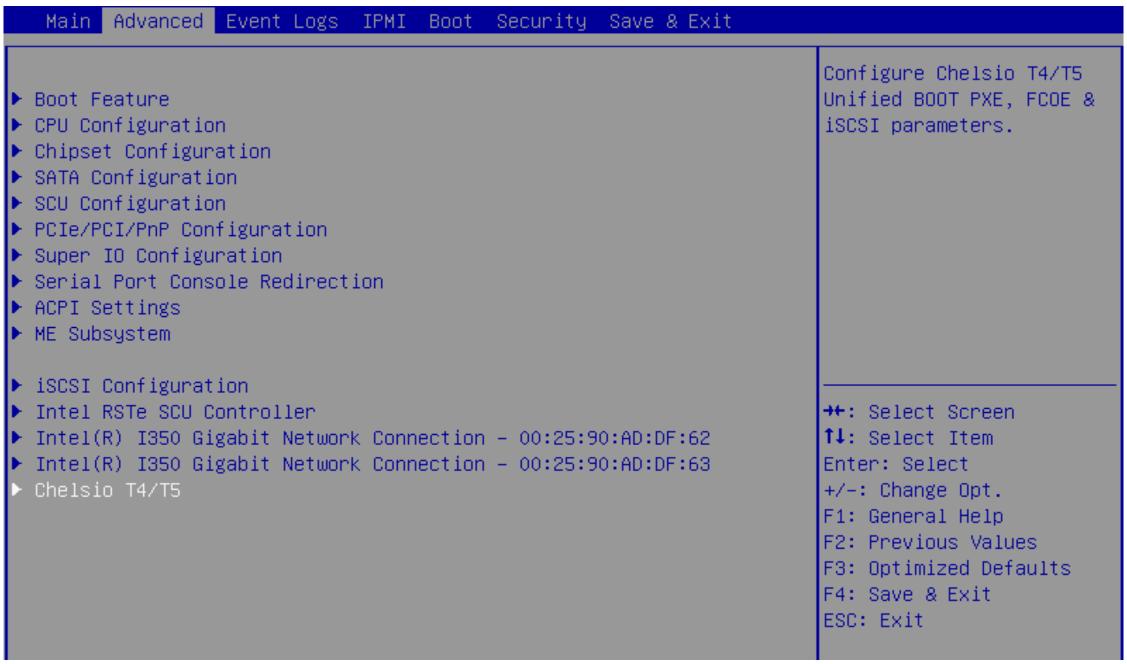
- *Only uEFI v2.1 and v2.3.1 supported.*
- *Any other uEFI version is NOT SUPPORTED and may render your system unusable.*

This section describes the method to perform iSCSI boot on uEFI platforms.

i. Reboot the system and go into BIOS setup.

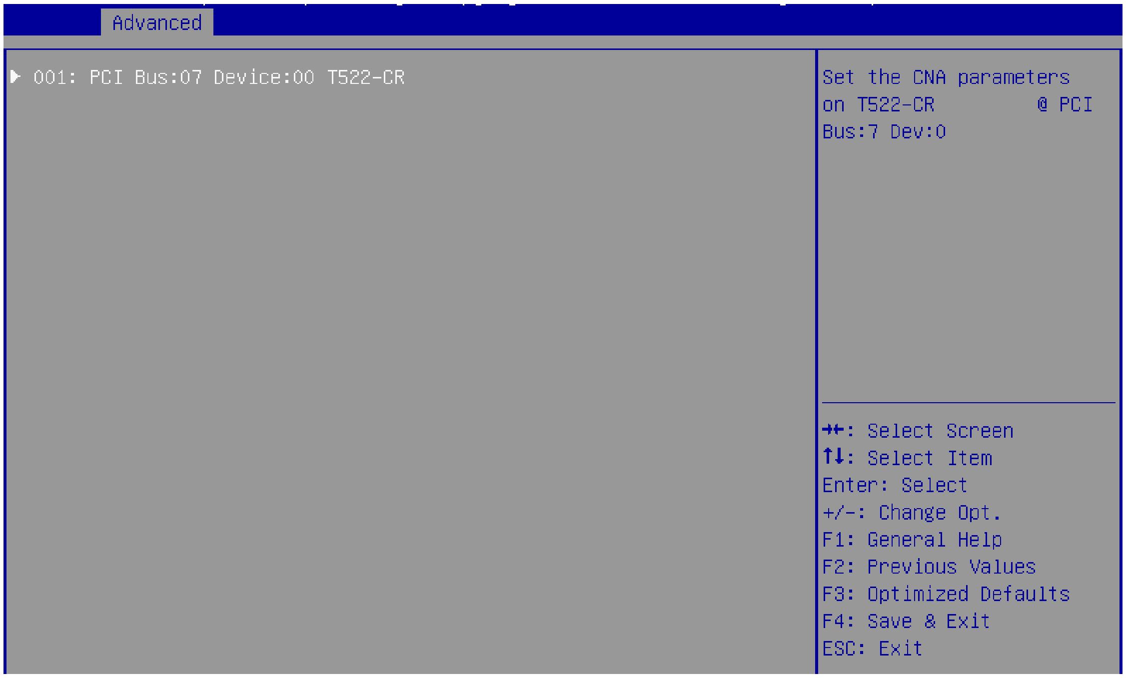
ii. Disable Secure Boot, if not already done.

iii. Select **Chelsio T4/T5** and press [Enter]

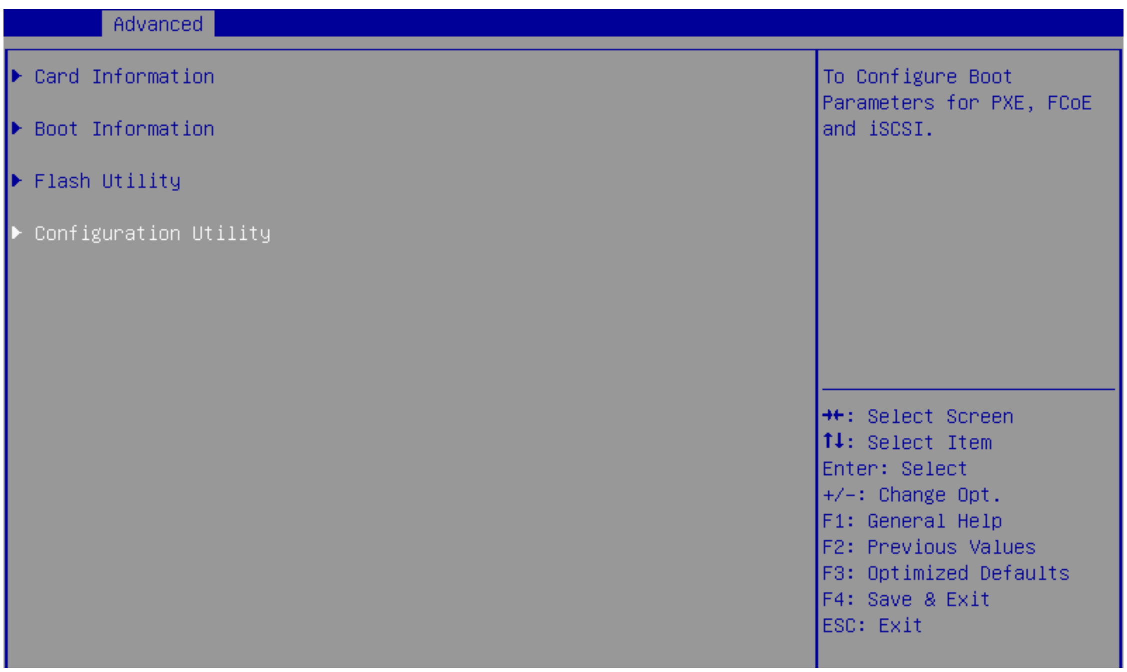


Note *If Chelsio T4/T5 is not listed, please ensure that Chelsio uEFI driver is loaded correctly as mentioned **here** in the **Flashing Firmware and Option ROM** section.*

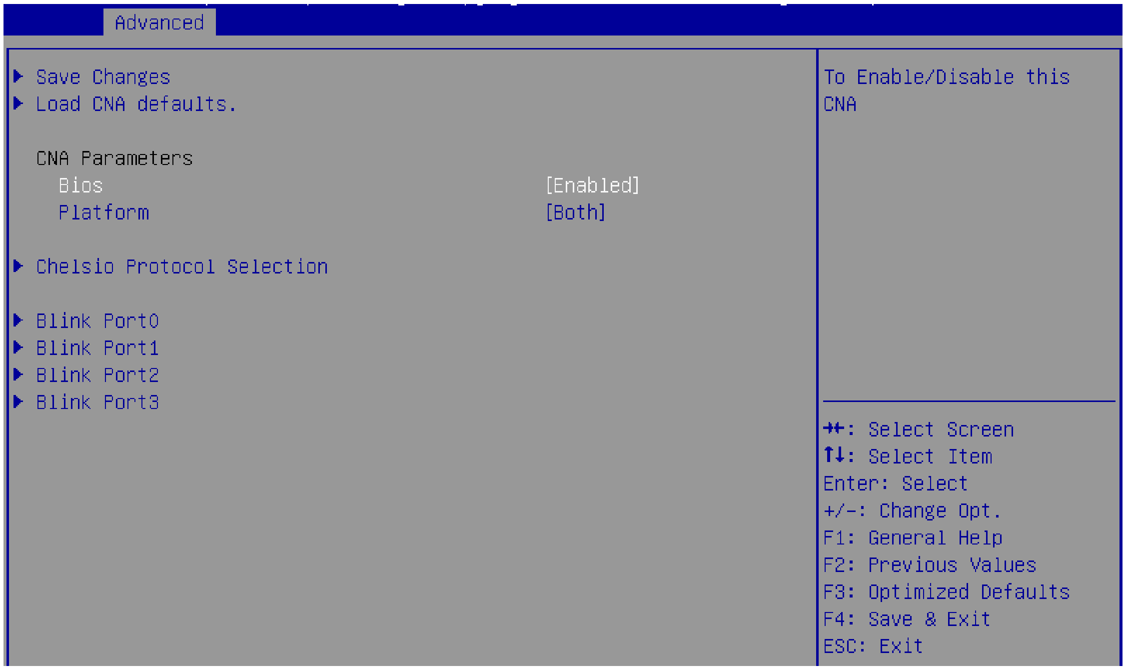
iv. Select the Chelsio adapter to be configured and press [Enter].



v. Select **Configuration Utility** and press [Enter].

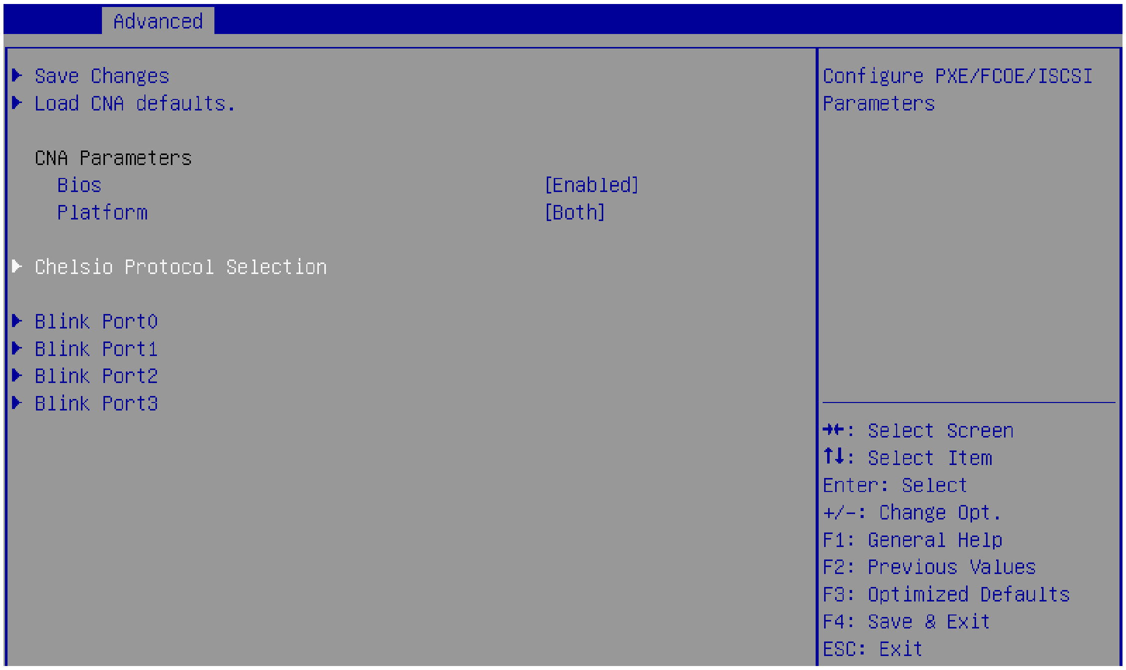


vi. Enable adapter BIOS if not already enabled.

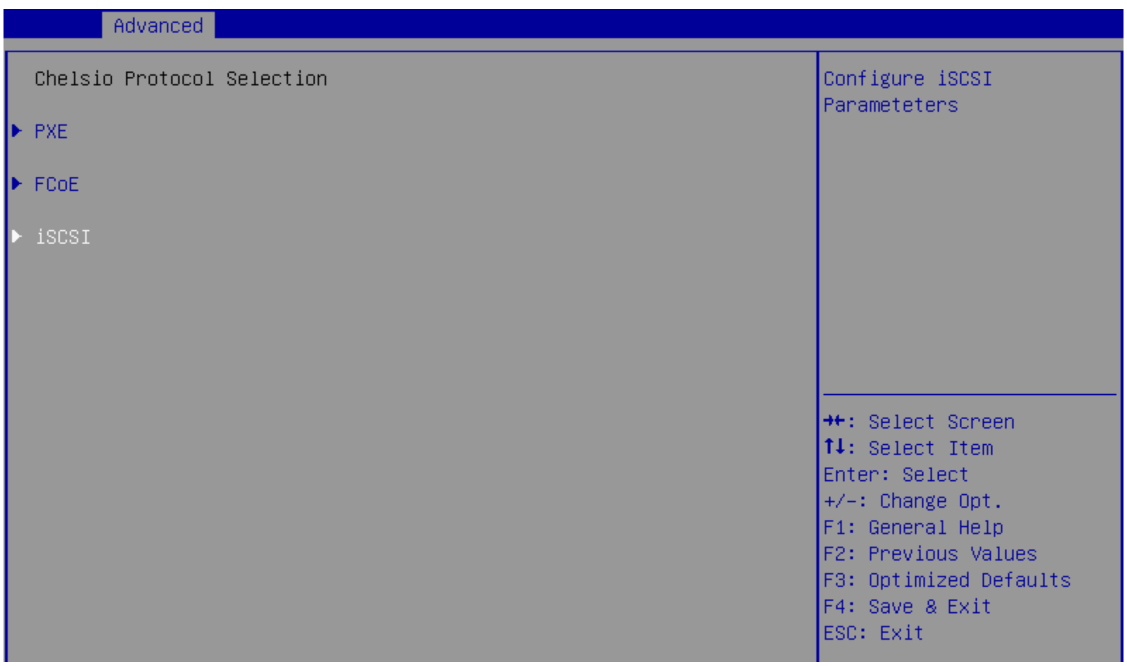


Note *It is highly recommended that you use the **Save Changes** option every time a parameter/option is changed.*

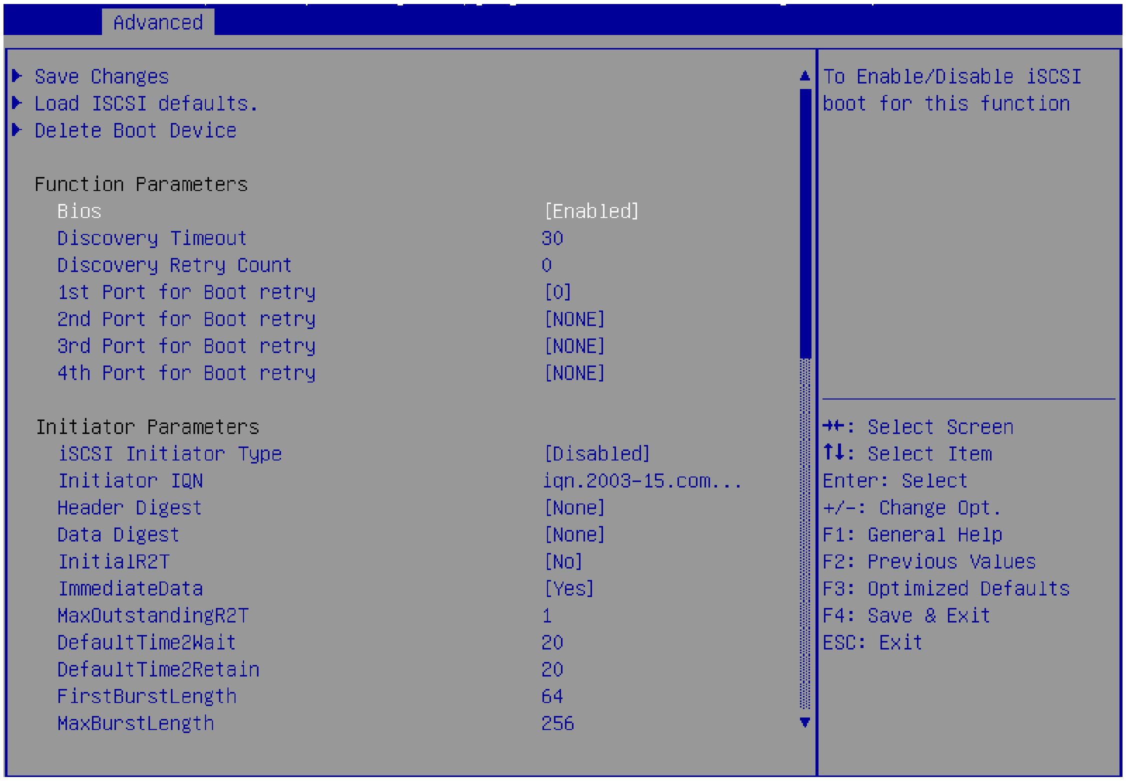
vii. Select **Chelsio Protocol Selection** and press [Enter].



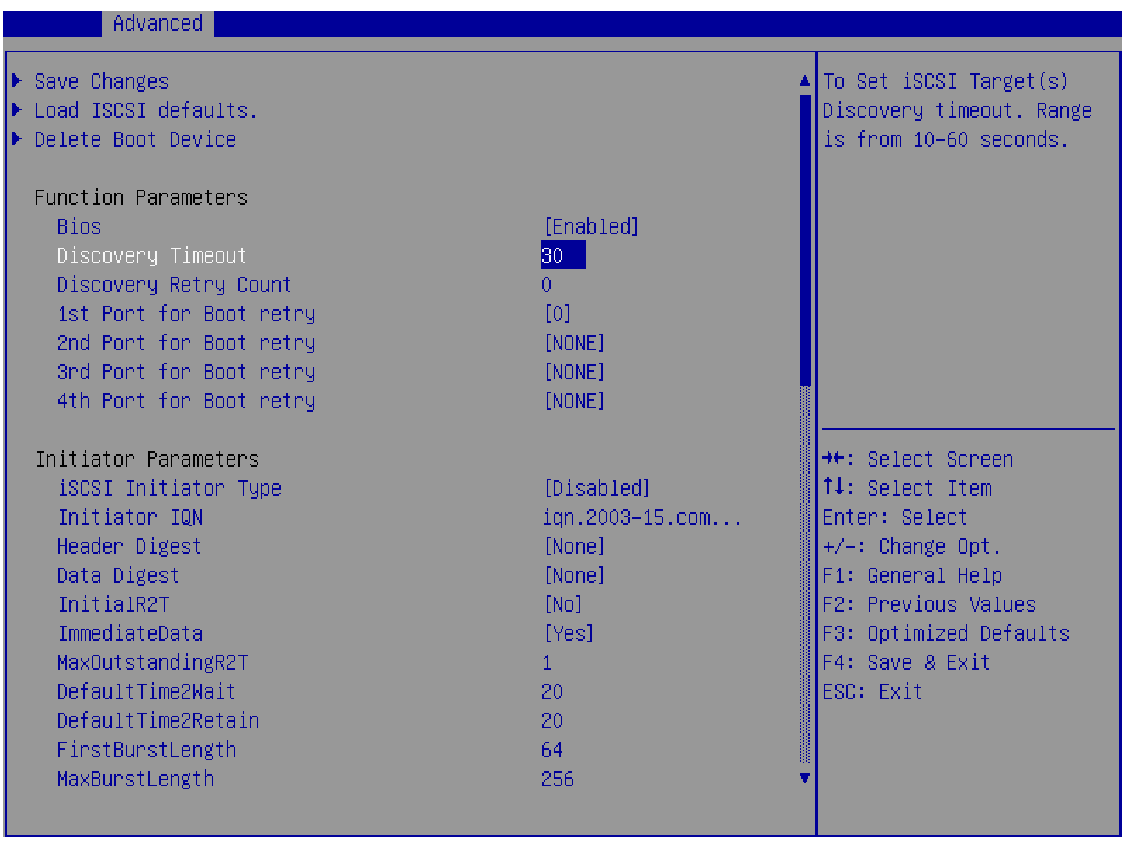
viii. Select **iSCSI** and press [Enter]



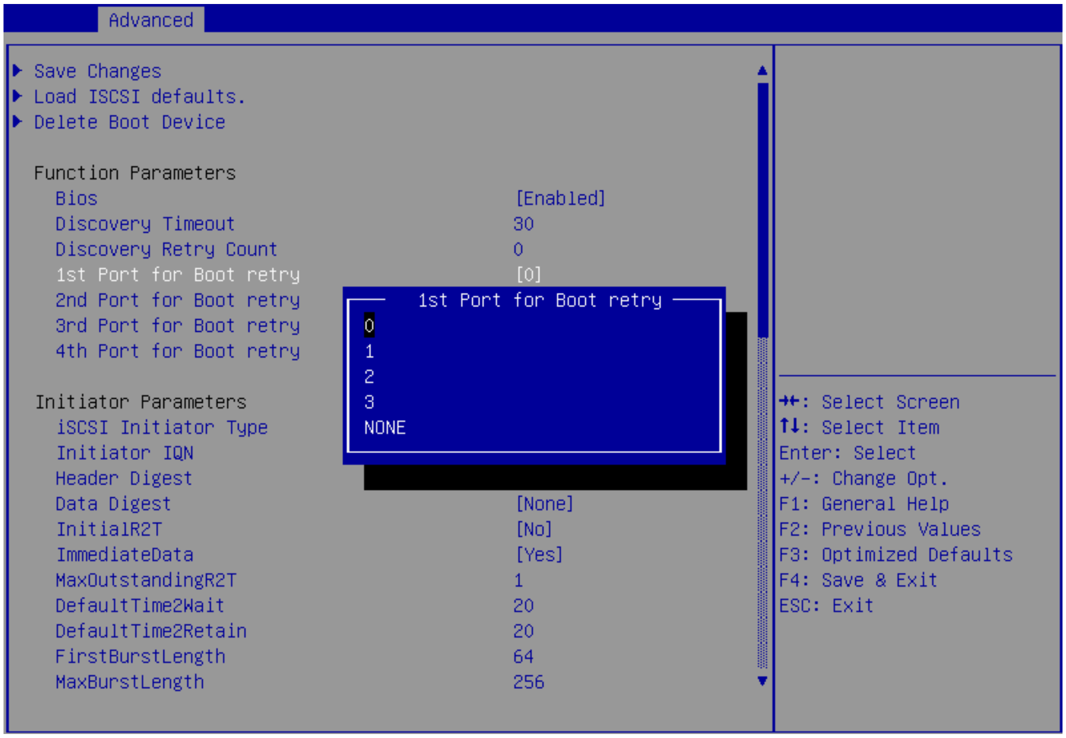
ix. Under **Function Parameters**, enable iSCSI BIOS, if not already enabled.



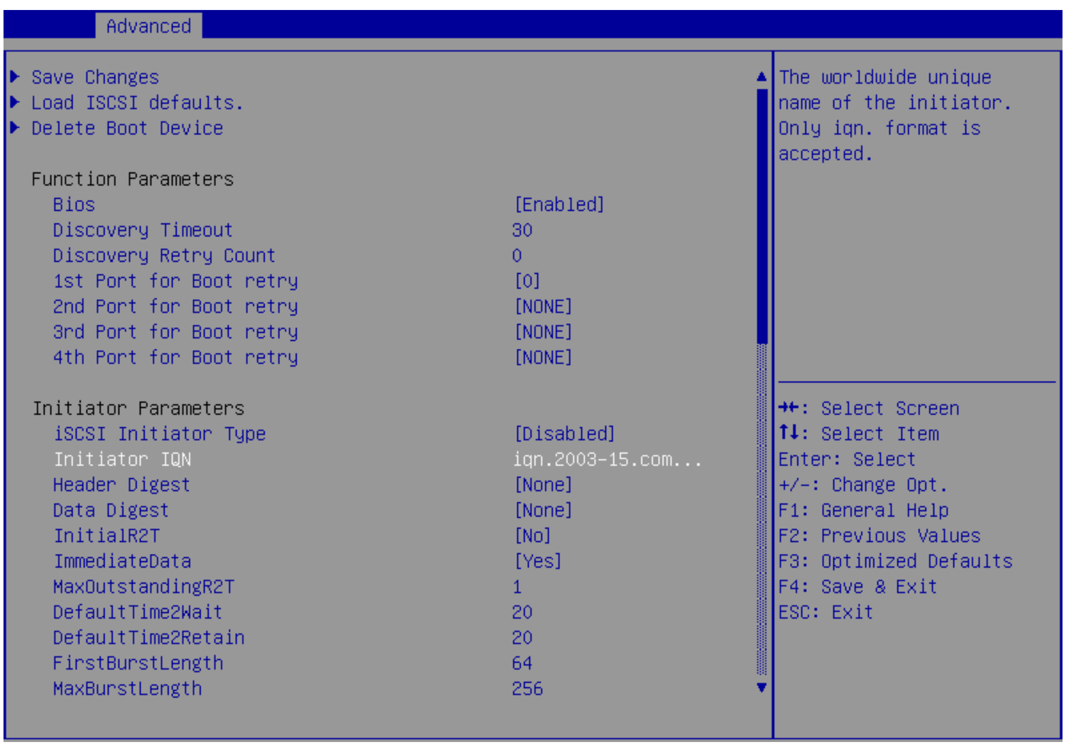
x. Set discovery timeout to a suitable value. Recommended value is >= 30



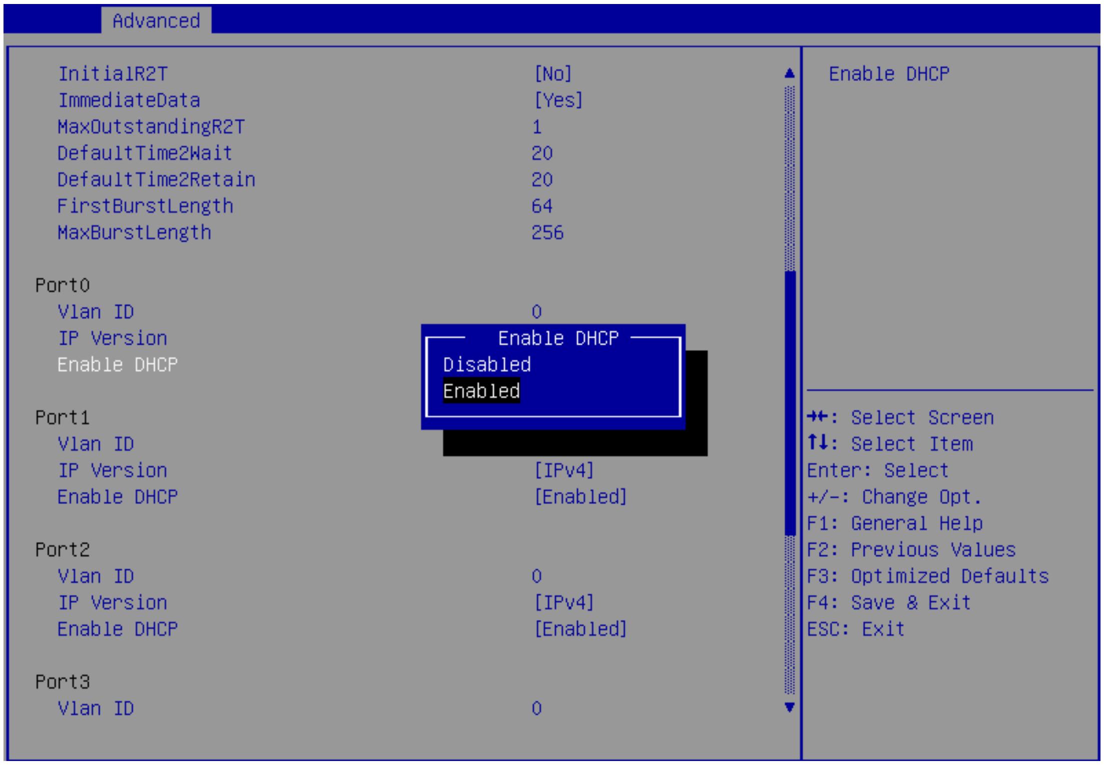
xi. Choose the order of the ports to discover iSCSI targets.



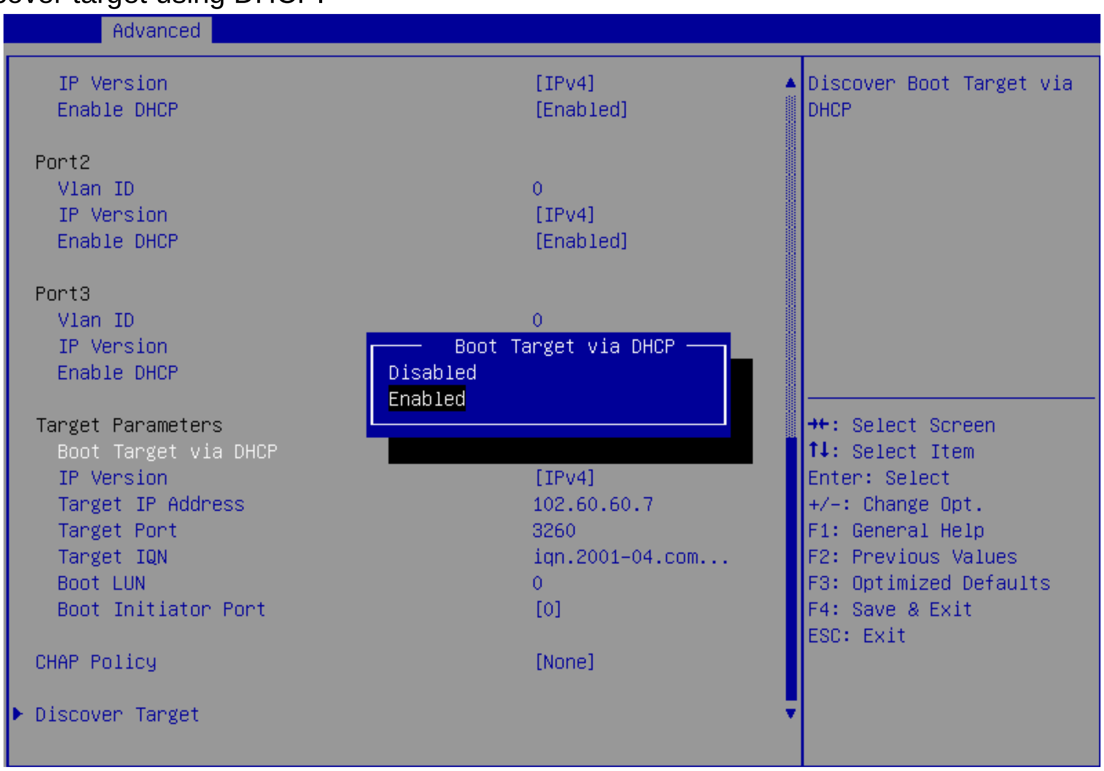
xii. Under **Initiator Parameters**, iSCSI Initiaitor properties like IQN, Header Digest, Data Digest, etc will be displayed. Change the values appropriately or continue with the default values.



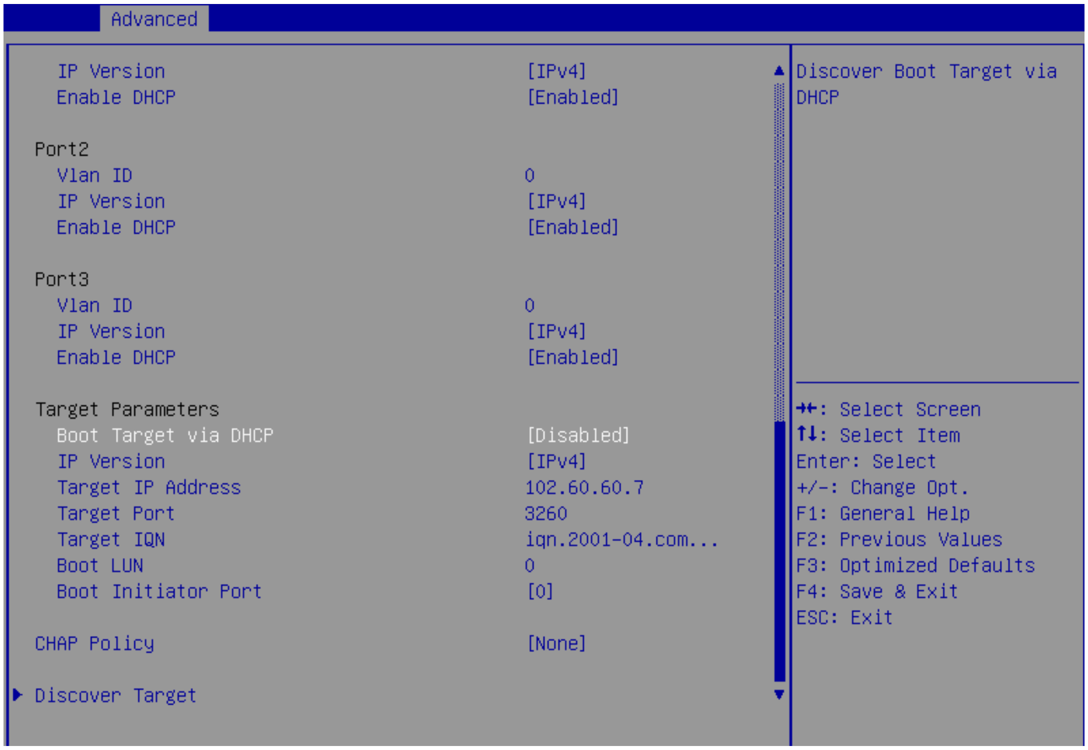
xiii. Under the first port, select **Enable DHCP** field, hit [Enter] and select **Enabled**. This will configure port using DHCP. Select **Disabled** to manually configure the port.



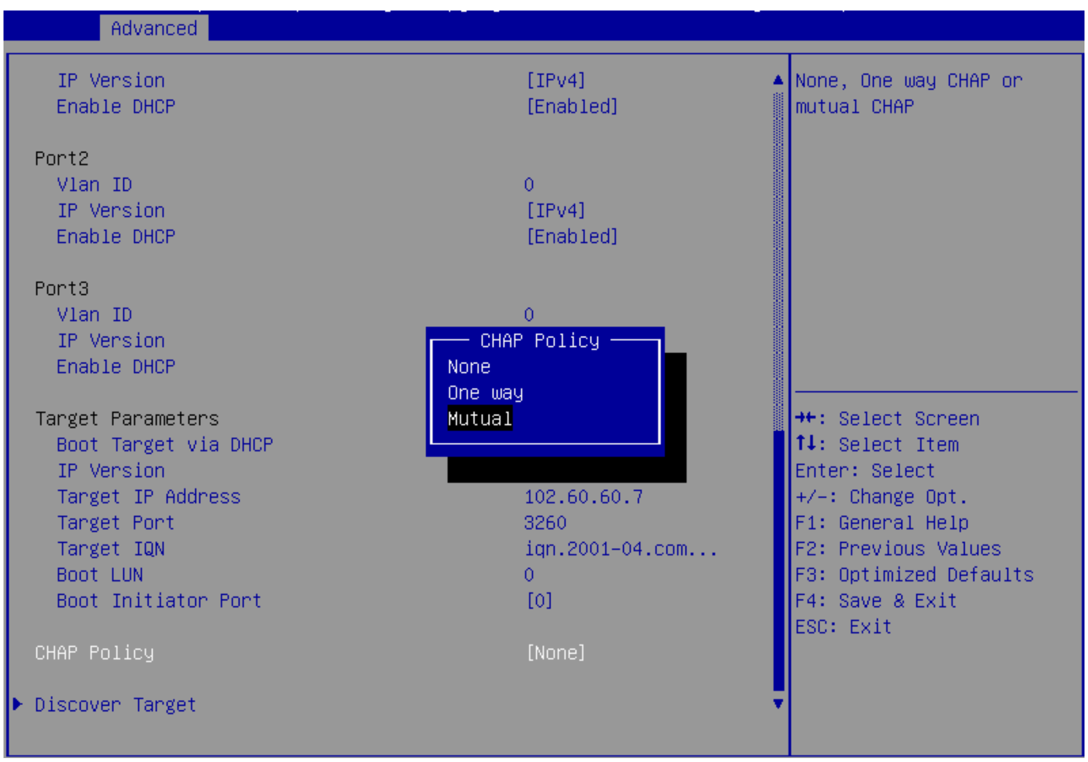
xiv. Under **Target Parameters**, select **Enabled** for the **Boot Target via DHCP** parameter to discover target using DHCP.



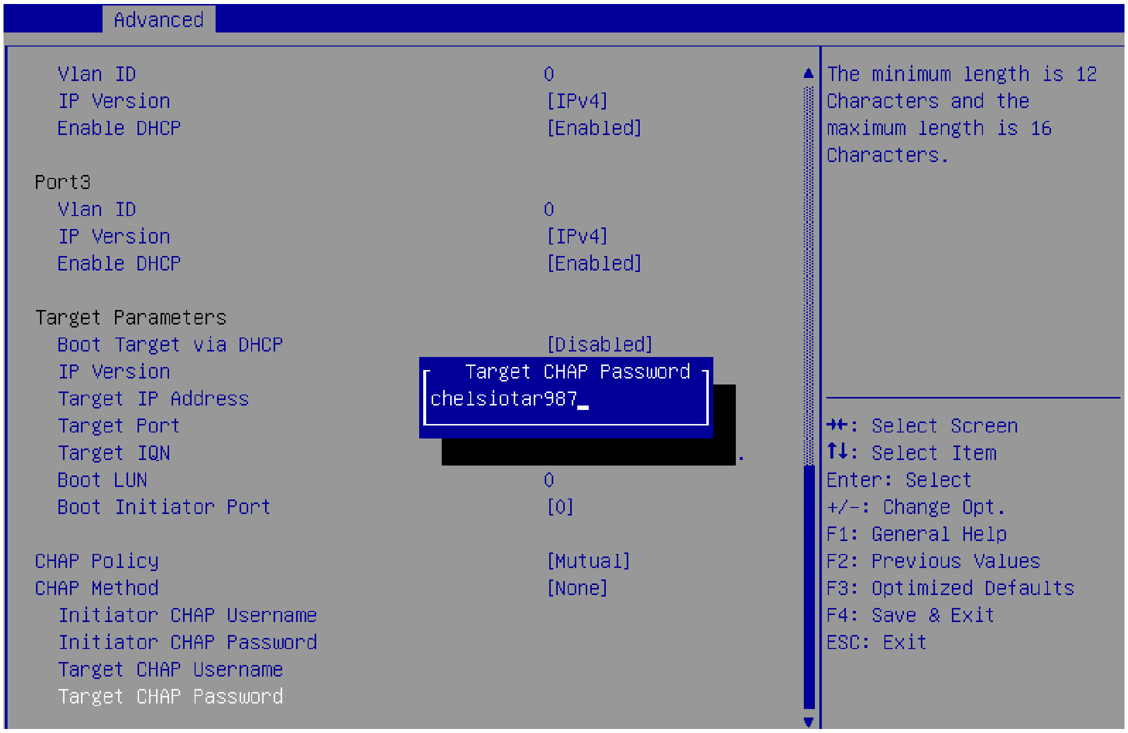
To discover target via static IP, select **Disabled** and provide the target IP.



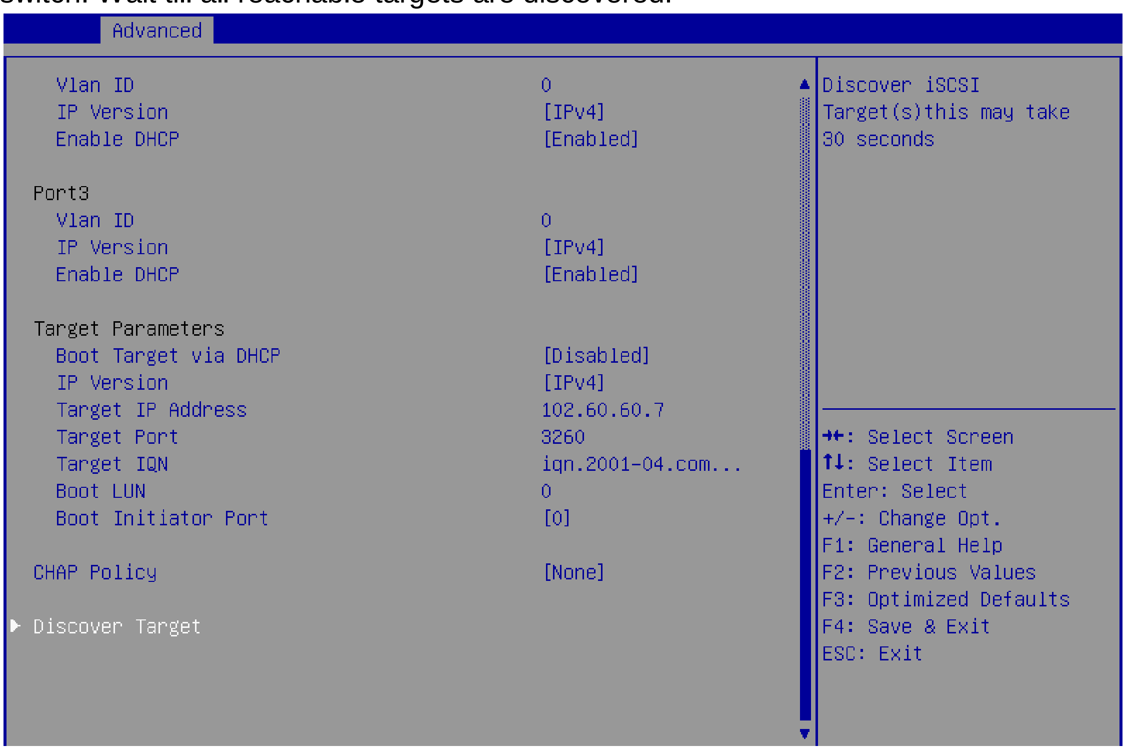
xv. CHAP authentication is disabled by default. To enable and configure, highlight **CHAP Policy** and hit [Enter]. Select the policy type from the corresponding pop-up and hit [Enter] again.



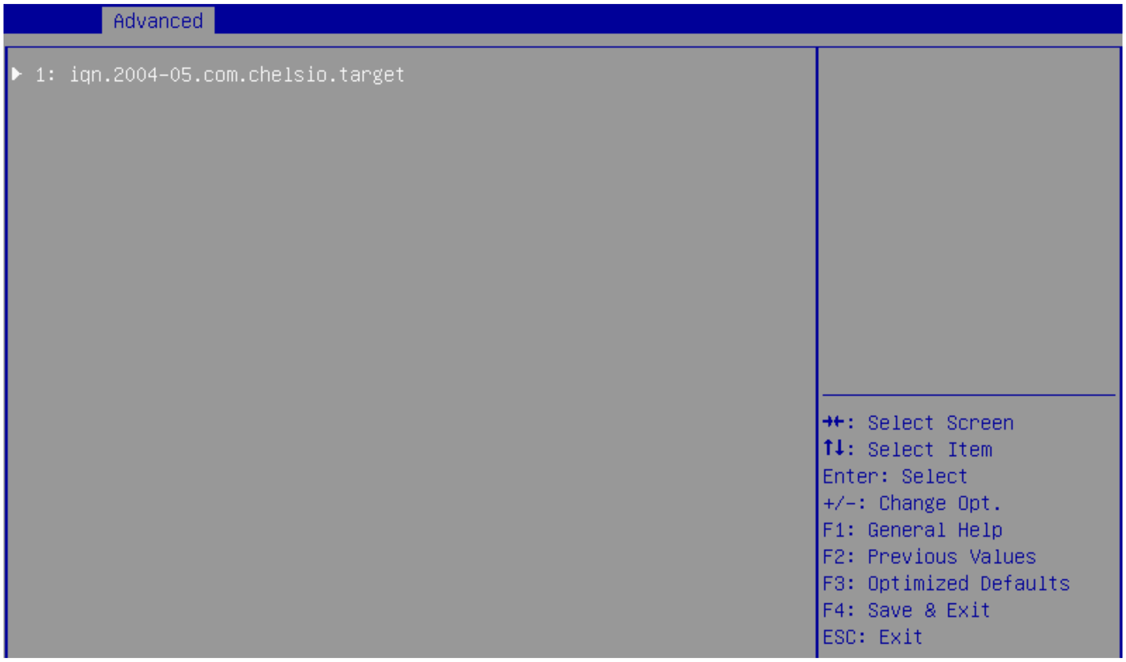
xvi. Provide Initiator and Target CHAP credentials according to the CHAP policy selected.



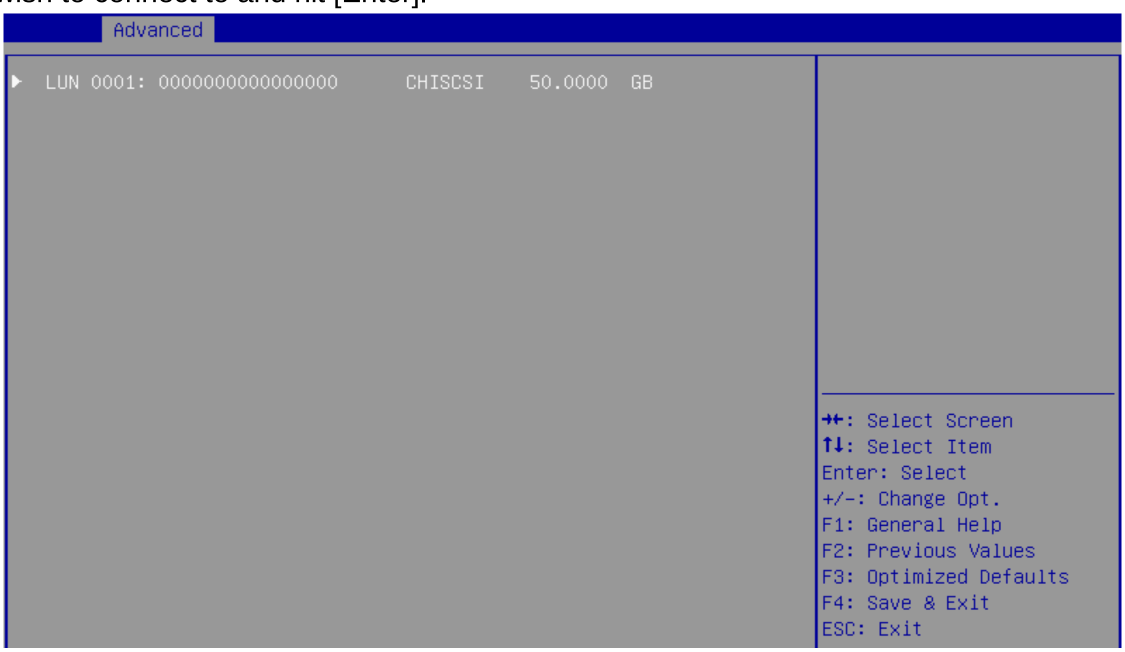
xvii. Select **Discover Target** and press [Enter] to discover iSCSI targets connected to the switch. Wait till all reachable targets are discovered.



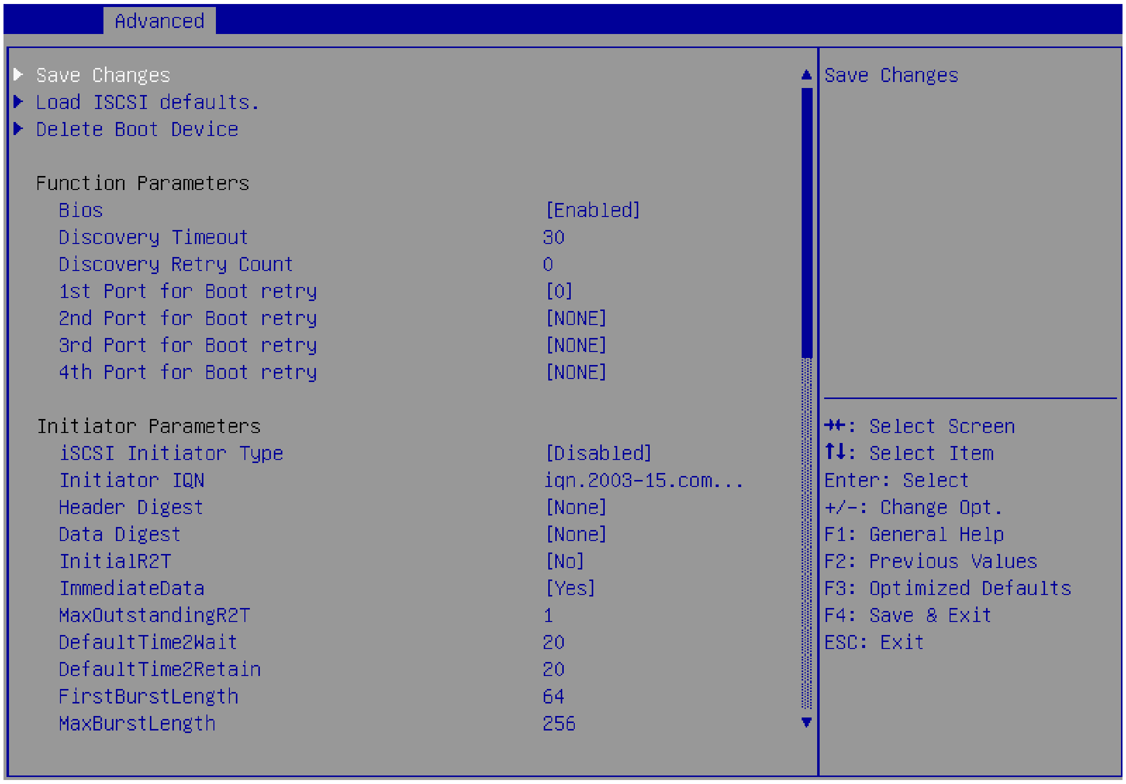
xviii. A list of available targets will be displayed. Select the target you wish to connect to and hit [Enter].



xix. A list of LUNs configured on the selected target will be displayed. Select the LUN you wish to connect to and hit [Enter].

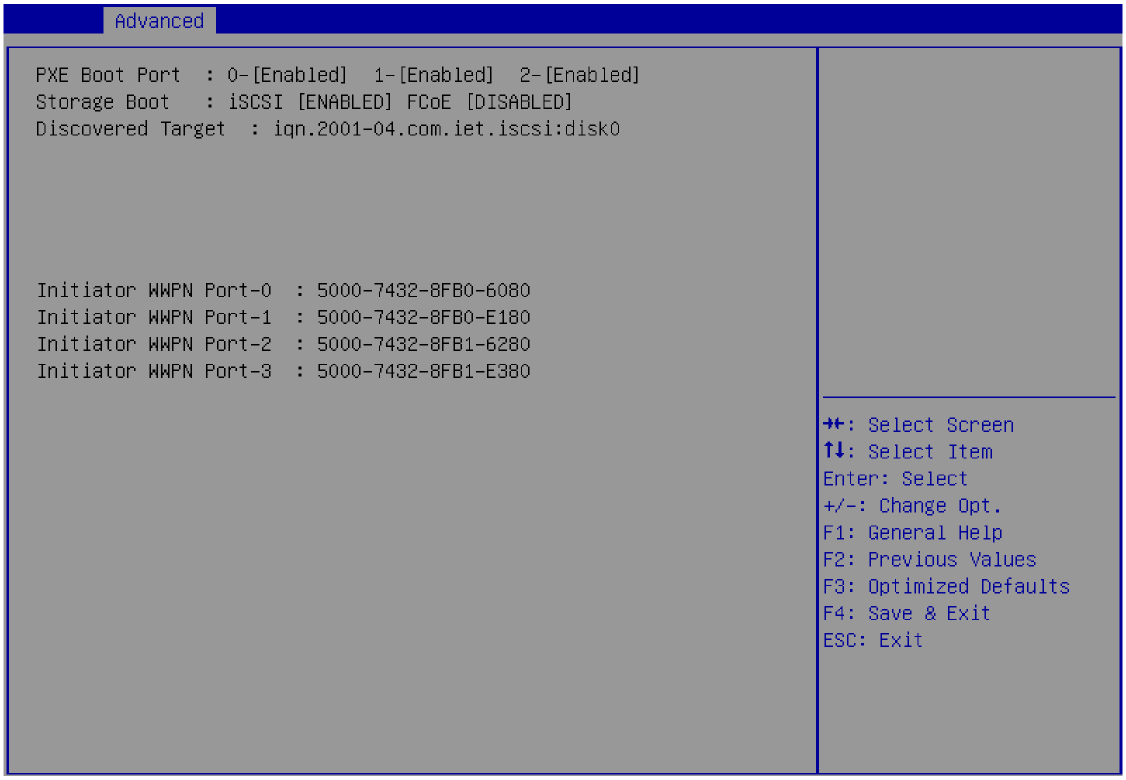


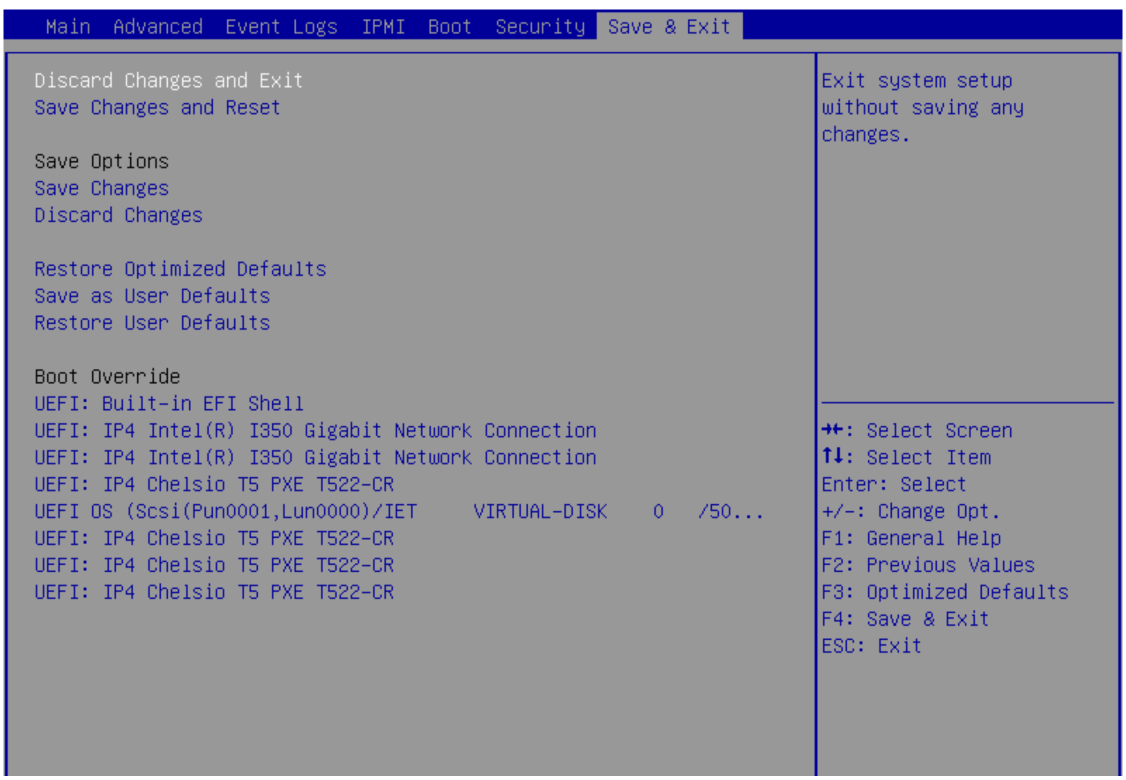
xx. Select **Save Changes** and press [Enter]



xx. Reboot the system for changes to take effect.

xxi. The discovered LUN should appear in the **Boot Configuration** section and system BIOS.





xxii. Select the LUN as the first boot device and exit from BIOS.

xxiii. Either boot from the LUN or install the required OS.

7. Creating Driver Update Disk (DUD)

The following section describes the procedure to create Driver Update Disks for RHEL and SLES distributions for Chelsio adapters. In case of T4 adapters, you can skip this step and use inbox drivers to install the operating system.

7.1. Creating DUD for RedHat Enterprise Linux

i. If you haven't done already, download the Chelsio Unified Wire driver package from Chelsio Download Center, service.chelsio.com

ii. Untar the package:

```
[root@host~]# tar zxvfm <driver_package>.tar.gz
```

iii. Change your current working directory to *LinuxDUD* directory:

```
[root@host~]# cd <driver_package>/Uboot/LinuxDUD
```

iv. Insert a blank, formatted USB flash drive.

v. Depending on the distribution to be installed, copy the corresponding image file to the USB drive. For example, execute the following command for RHEL 6.6:

```
[root@host~]# cp Chelsio-DriverUpdateDisk-RHEL6.6-x86_64-x.xx.x.x.img <path
to USB drive>
```

Note *For RHEL 7.X, use Chelsio-DriverUpdateDisk-RHEL7.X-x86_64-x.xx.x.x.iso*

7.2. Creating DUD for Suse Enterprise Linux

i. If you haven't done already, download Chelsio Unified Wire driver package from Chelsio Download Center, service.chelsio.com

ii. Untar the package,

```
[root@host~]# tar zxvfm <driver_package>.tar.gz
```

iii. Insert a blank USB flash drive.

iv. Format the USB drive

```
[root@host~]# mkfs.vfat /dev/sda1
```

v. Depending on the distribution to be installed, copy the corresponding image file to the USB stick. For example, execute the following command for SLES 11 sp4.

```
[root@host~]# dd if=/root/<driver_package>/Uboot/LinuxDUD/Chelsio-
DriverUpdateDisk-SLES11sp4-x86_64-x.xx.x.x.img of=/dev/sda1
```

8. OS Installation

8.1. Installation using Chelsio DUD

This is the recommended method for installing Linux OS using Chelsio PXE boot. The Chelsio Driver Update Disk (DUD) has support for all the new adapters. Use Network Boot (PXE Boot) media to install the OS, and provide the Driver Update Disk as per the detailed instructions for each OS.

The DUD supports installation of Linux distributions using Chelsio adapters over Network. There may be built-in Chelsio driver in these distributions. The driver may or may not work with Chelsio adapters, depending on the adapter in use, and the version of the driver that shipped in that particular distribution. Please flash the firmware provided in the package.

8.1.1. RHEL 7.X installation

i. Please make sure that the USB drive with DUD image is inserted. Type *e* and then *dd* at the boot prompt for the installation media. The *dd* option specifies that you will be providing a Driver Update Disk during the installation.

ii. You will be asked to select the Driver Update Disk device from a list. USB drives usually show up as SCSI disks in Linux. Enter the index number of the device to be used and hit [Enter].

iii. The installer will search and display DUD image files found in the selected device. Enter the index number of the file to be used and hit [Enter].

iv. Drivers provided in the DUD will be listed. Enter 1 to select FCoE driver (*csiostor*), or 2 to select Network driver (*cxgb4*). Hit [Enter]

```
Page 1 of 1
Select drivers to install
  1) [ ] /media/DD//rpms/x86_64/kmod-csiostor-[redacted].rpm
  2) [ ] /media/DD//rpms/x86_64/kmod-cxgb4-[redacted].rpm

# to toggle selection, 'n'-next page, 'p'-previous page or 'c'-continue: 1
```

v. To select the next driver, enter the driver index or enter "C" to start the loading process. Hit [Enter]. The selected driver(s) will now be loaded.

```
Page 1 of 1
Select drivers to install
  1) [x] /media/DD//rpms/x86_64/kmod-csiostor-[redacted].rpm
  2) [ ] /media/DD//rpms/x86_64/kmod-cxgb4-[redacted].rpm

# to toggle selection, 'n'-next page, 'p'-previous page or 'c'-continue: 2
```

Note *To deselect a driver, enter the index of the selected driver and hit [Enter]*

vi. The **Driver disk prompt** will be displayed again. Follow the same procedure mentioned above to select any other drivers you wish to load or press "C" to skip and start the loading process.

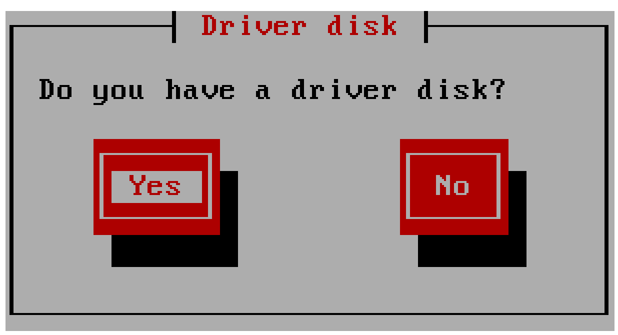
vii. After the drivers are successfully loaded, OS installation will commence. Proceed as usual.

8.1.2. RHEL 6.X installation

i. Please make sure that the USB drive with DUD image is inserted. Press *Tab* and then type *dd* at the boot prompt for the installation media. The *dd* option specifies that you will be providing a Driver Update Disk during the installation.

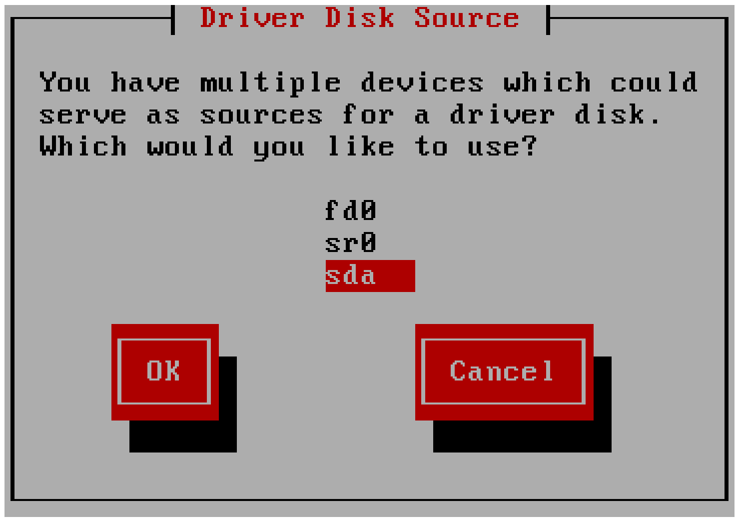
Note *In case of iSCSI boot, type **dd ip=ibft***

ii. The installer will load and prompt you for the driver update disk. Select "Yes" and hit [Enter] to proceed.

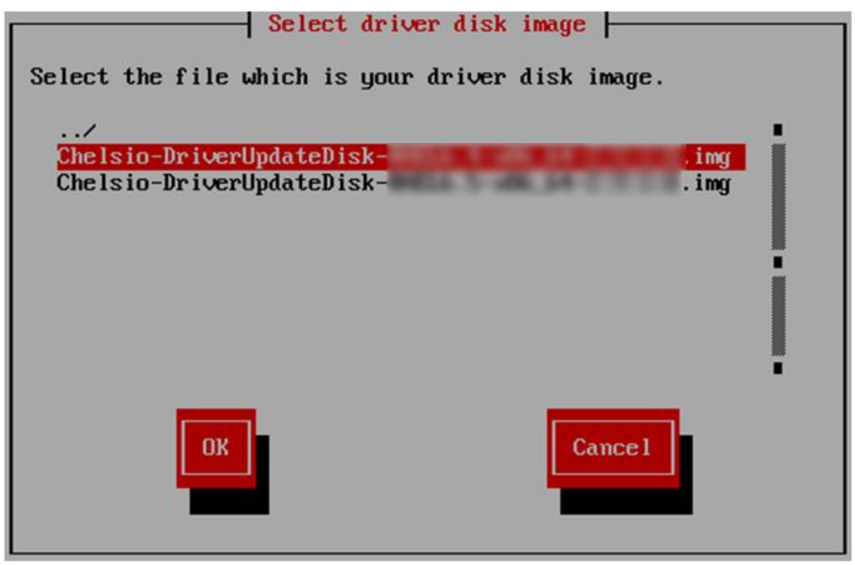


iii. You will be asked to select the Driver Update Disk device from a list. USB drives usually show up as SCSI disks in Linux. So if there are no other SCSI disks connected to the system, the USB drive would assume the first drive letter “a”. Hence the drive name would be “sda”.

You can view the messages from the Linux kernel and drivers to determine the name of the USB drive, by pressing [Alt] + [F3] or [Alt] + [F4]. Press [Alt] + [F1] to get back to the list.

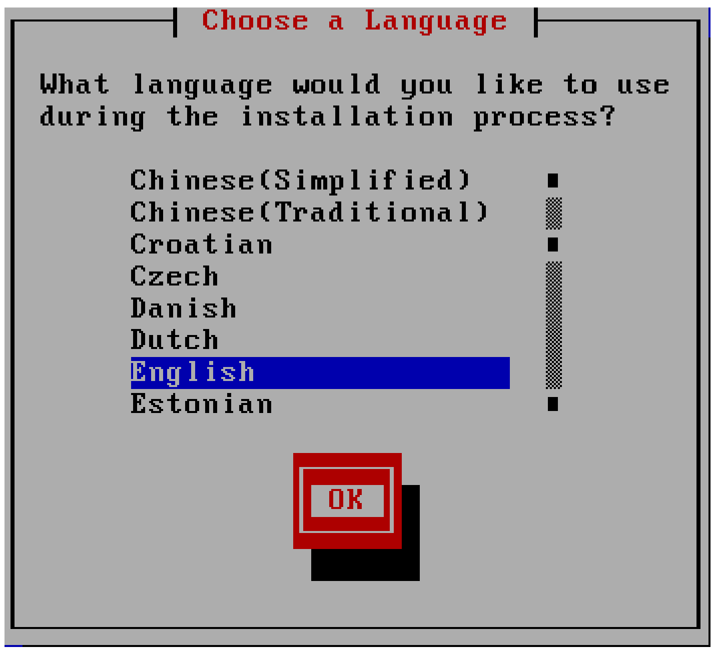


iv. Select the Appropriate image file and Choose “OK”. Now the installer will search for the appropriate drivers from the driver disk and load them. This step may take some time. Check on the [Alt] + [F3] or [Alt] + [F4] screens for log messages.

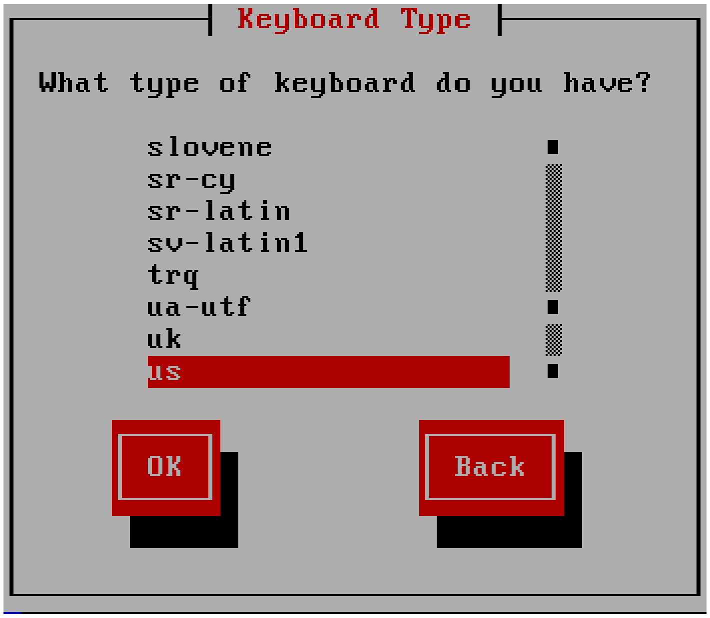


v. The installer will ask if you wish to load more drivers. Choose “Yes” to load if you have any other drivers to load. Otherwise choose “No”.

vi. Select the required language from the list.



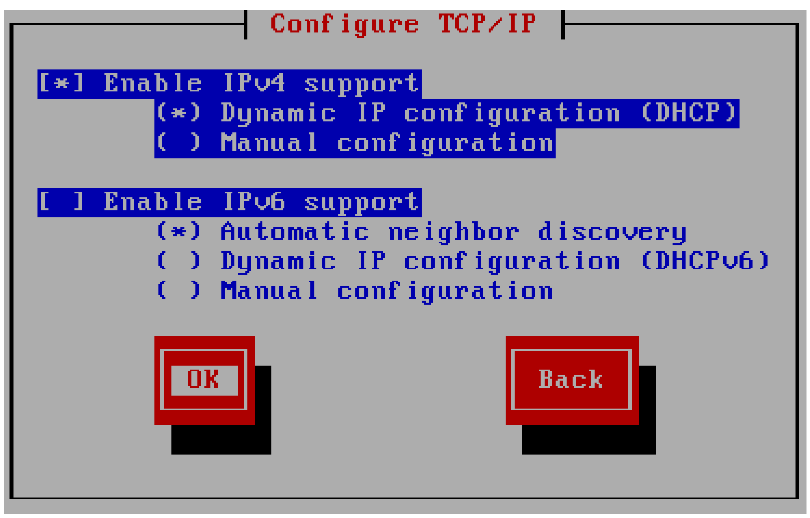
vii. Select the type of keyboard you have from the list.



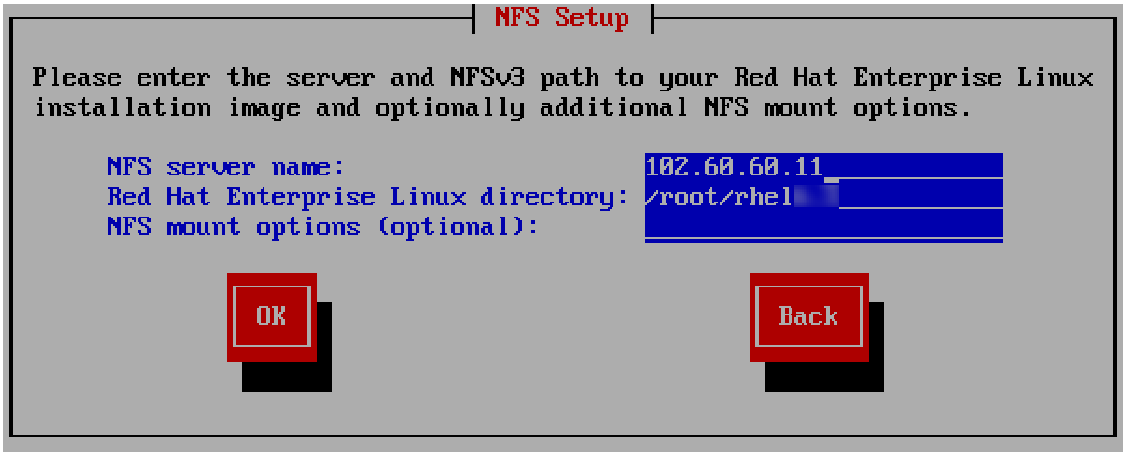
viii. In this step, you can choose the source which contains the OS installation ISO image. In this case, select “NFS directory”.

ix. The Chelsio detwork devices will be displayed. Select the appropriate Chelsio NIC interface to proceed with installation.

x. Here you can specify if you want to configure your network interfaces using DHCP or manually using IPv4. IPv6 is currently not supported. Hence disable IPv6 before proceeding.

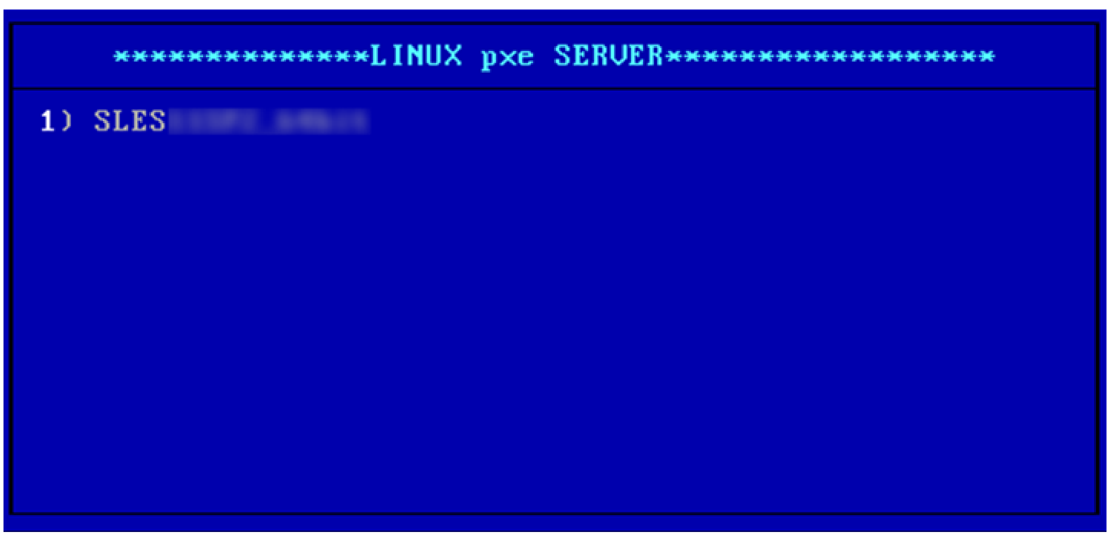


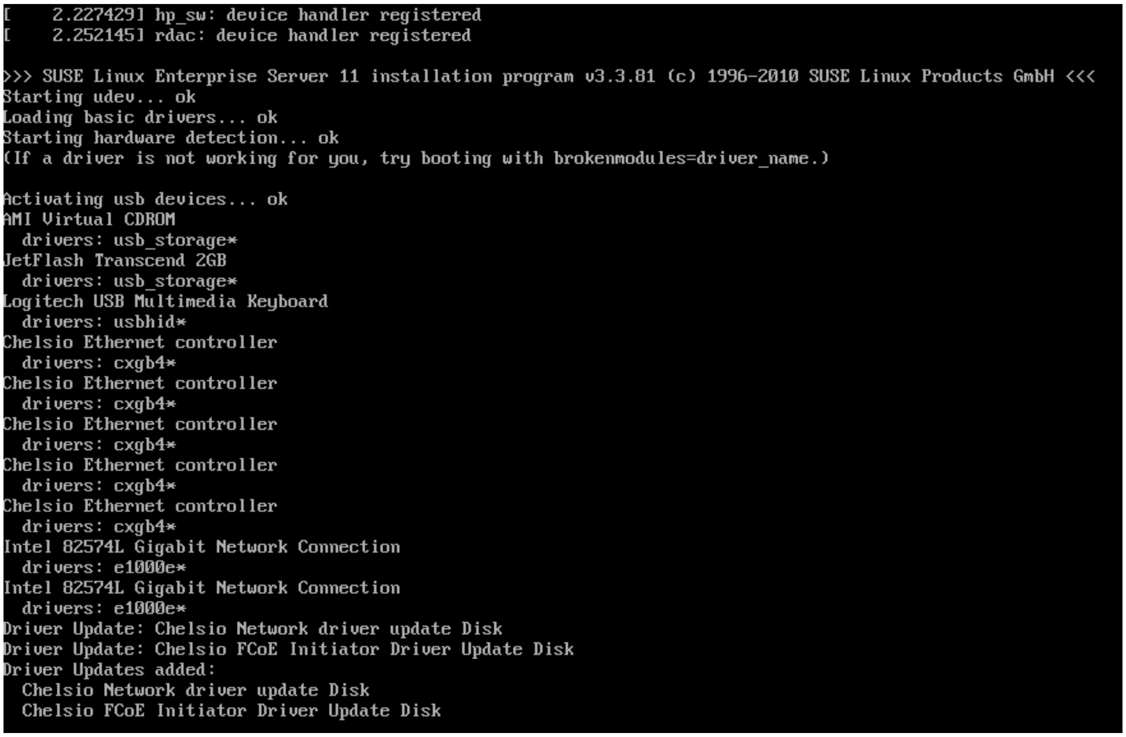
xi. Proceeding with the installation will get NFS/FTP/HTTP setup page. Here, provide NFS server details to proceed with the installation. Then the graphical Installation screens for RHEL will appear. Proceed with the installation as usual.



8.1.3. SLES 11 SPx/SLES 12/SLES 12 SPx installation

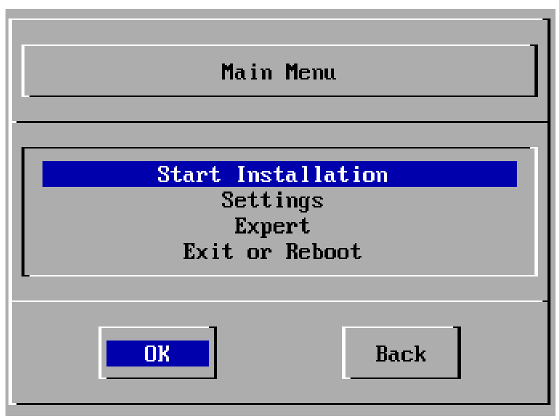
i. Please make sure that the USB drive with DUD image is inserted.
ii. Select the appropriate entry from the PXE menu and press [Enter].

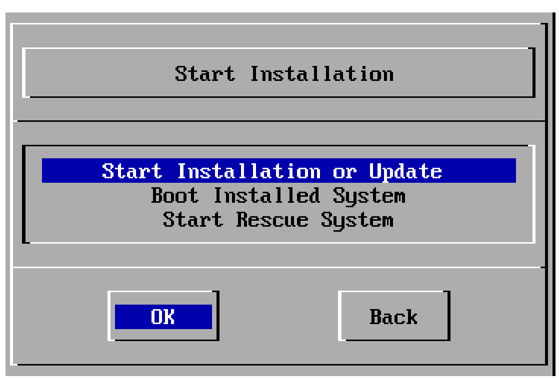




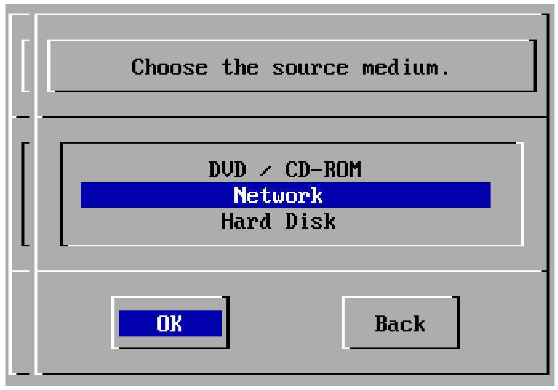
```
[    2.227429] hp_sw: device handler registered
[    2.252145] rdac: device handler registered

>>> SUSE Linux Enterprise Server 11 installation program v3.3.81 (c) 1996-2010 SUSE Linux Products GmbH <<<
Starting udev... ok
Loading basic drivers... ok
Starting hardware detection... ok
(If a driver is not working for you, try booting with brokenmodules=driver_name.)

Activating usb devices... ok
AMI Virtual CDROM
  drivers: usb_storage*
JetFlash Transcend 2GB
  drivers: usb_storage*
Logitech USB Multimedia Keyboard
  drivers: usbhid*
Chelsio Ethernet controller
  drivers: cxgb4*
Chelsio Ethernet controller
  drivers: cxgb4*
Chelsio Ethernet controller
  drivers: cxgb4*
Chelsio Ethernet controller
  drivers: cxgb4*
Chelsio Ethernet controller
  drivers: cxgb4*
Intel 82574L Gigabit Network Connection
  drivers: e1000e*
Intel 82574L Gigabit Network Connection
  drivers: e1000e*
Driver Update: Chelsio Network driver update Disk
Driver Update: Chelsio FCoE Initiator Driver Update Disk
Driver Updates added:
  Chelsio Network driver update Disk
  Chelsio FCoE Initiator Driver Update Disk
```

iii. Select “Start Installation” and then “Start Installation or Update”.

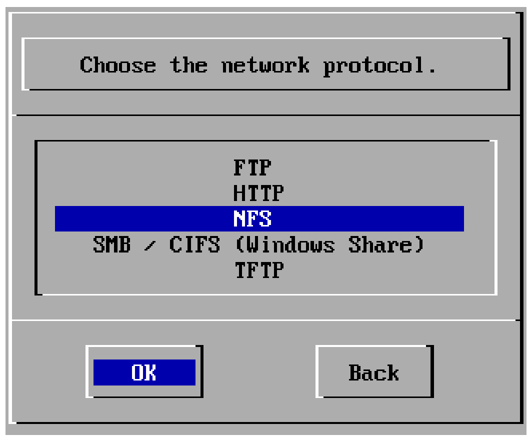




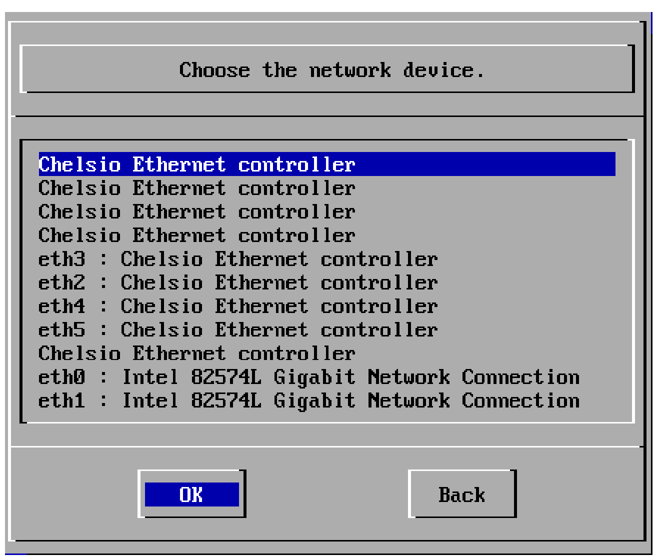
iv. Select “Network” as the source of medium to install the SLES Operating System.



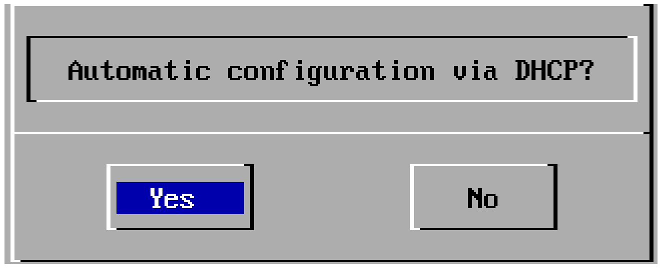
v. Select the desired Network protocol from the list presented.



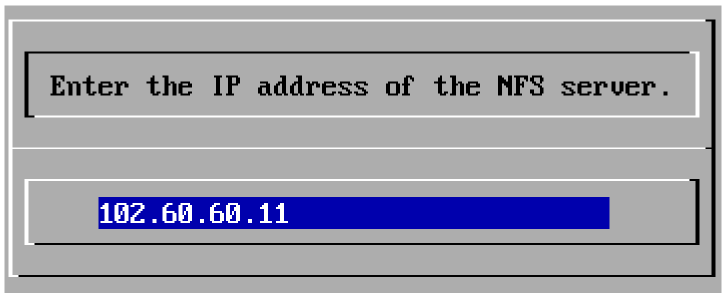
vi. Select the appropriate Chelsio interface from the list to proceed with installation. You can view the messages from the Linux kernel and drivers to determine the name of NIC interface by pressing [Alt] + [F3] or [Alt] + [F4]. Press [Alt] + [F1] to get back to the list.



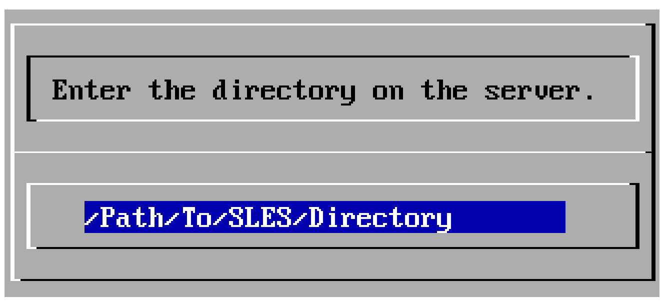
vii. Select “Yes” to configure the network interface selected in the previous step using DHCP.



viii. Provide a valid NFS/FTP/HTTP/TFTP Server IP address to proceed.



ix. Provide a valid directory path to the operating system to be installed.



x. Proceed with the installation as usual.

8.2. Installation on FCoE LUN

- If you are installing using CD/DVD, please make sure that the USB drive with DUD image is inserted. Also, change the boot priority to boot from CD/DVD in the BIOS setup.
  - i. Insert the OS installation disc into your CD/DVD ROM.
  - ii. On the Grub menu, choose *Install or upgrade an existing system* option if not already selected.
  - iii. Type *e* and then *dd* at the boot prompt for RHEL 7. For RHEL 6 and SLES distributions, press *Tab* and then type *dd.*

iv. Load Chelsio Driver Update Disk depending on the Linux distribution (**Click here** for RHEL 7.x; **Click here** for RHEL 6.x; **Click here** for SLES 11 SPx/SLES 12/SLES 12 SPx).

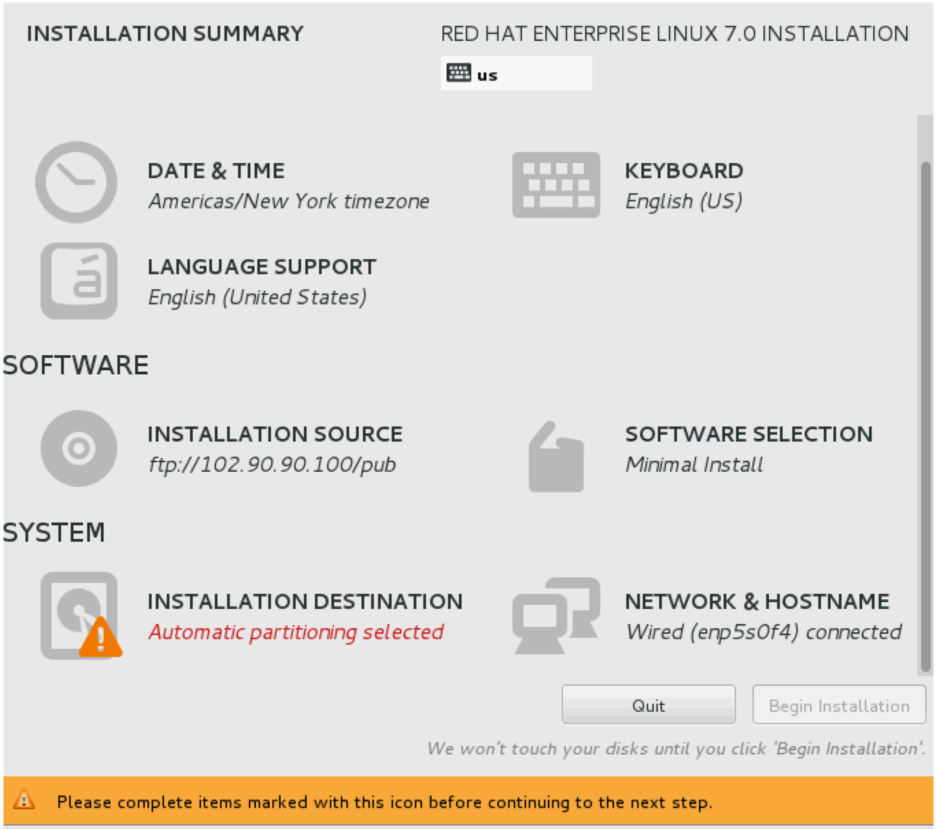
- If you are installing from a PXE server, please refer **8.1. Installation using Chelsio DUD** (**Click here** for RHEL 7.x; **Click here** for RHEL 6.x; **Click here** for SLES 11 SPx/ SLES 12/SLES 12 SPx) section to load Chelsio Driver Update Disk.

After successfully loading Chelsio DUD, follow the procedure mentioned below to continue installation, based on the distribution.

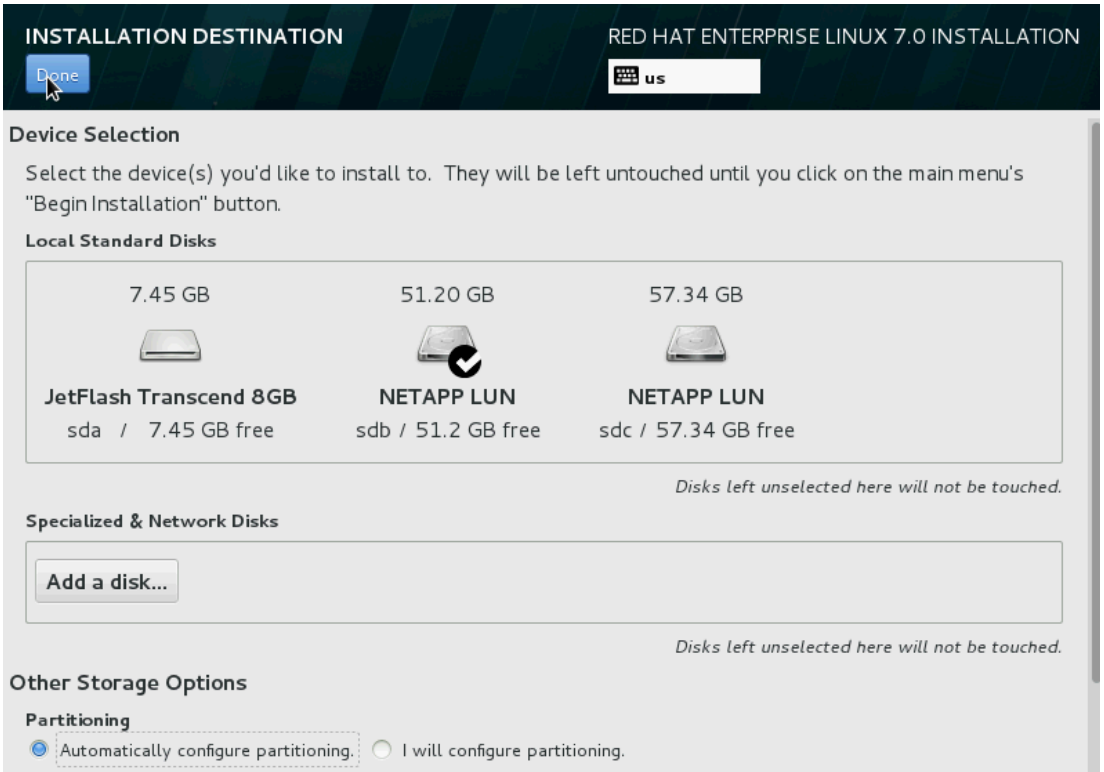
8.2.1. RHEL 7.x

i. Choose your installation language and click **Continue**

ii. Click **INSTALLATION DESTINATION** under **SYSTEM.**

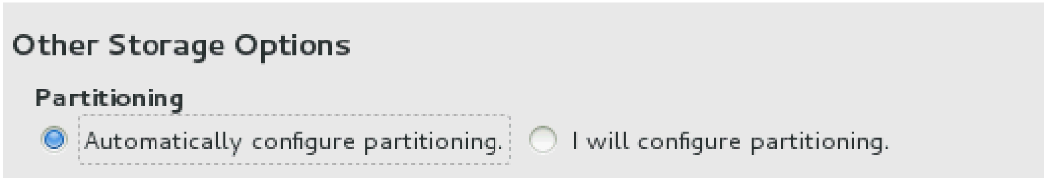


iii. The discovered FC/FCoE LUNs will appear as local storage in the **Local Standard Disks** section**.** Select the LUN which was saved as boot device in system BIOS.



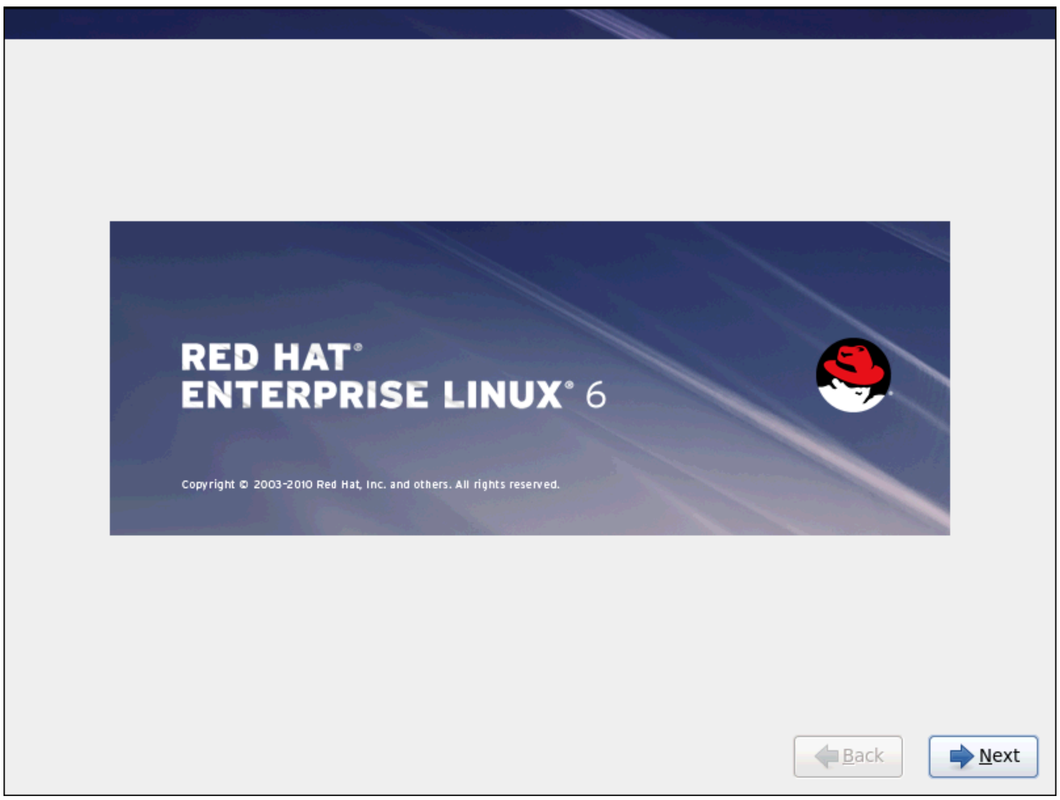
Note *Make sure the same LUN discovered at the Option ROM stage is selected for OS installation.*

iv. Under **Other Storage Options**, you can either chose to configure partition automatically or manually. Select the appropriate option and click **Done**. Then proceed with the installation as usual.

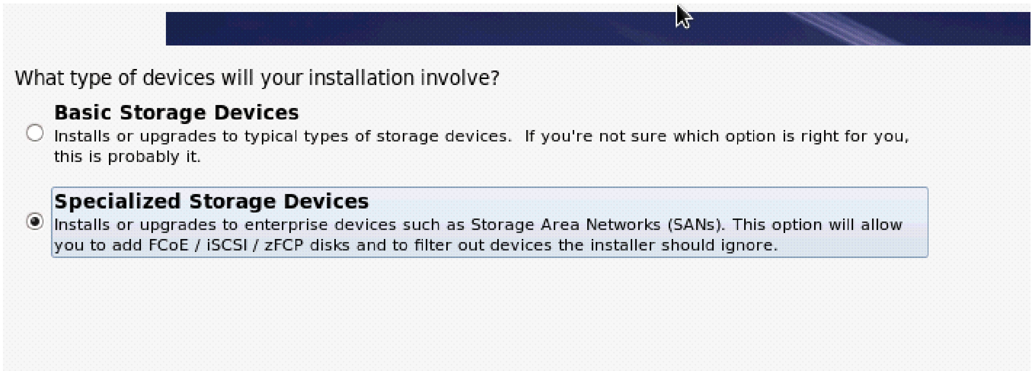


8.2.2. RHEL 6.x Installation

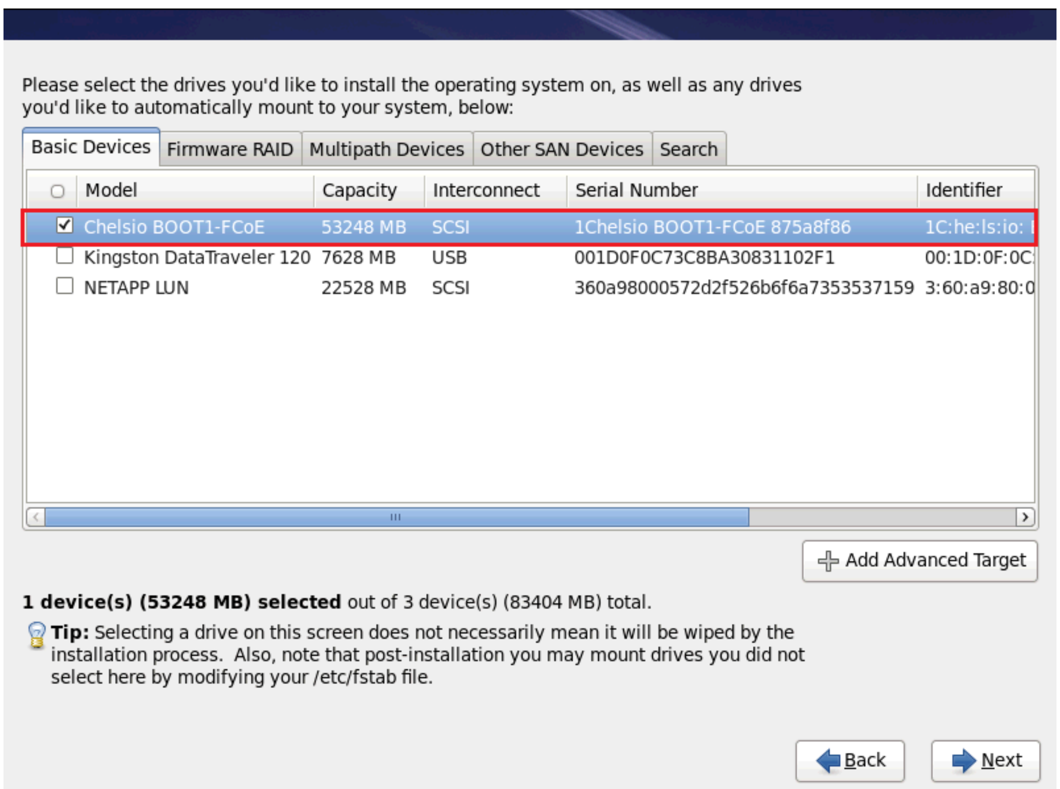
i. Click **Next** when the graphical installer screen appears.



ii. Select **Specialized Storage Devices** radio button and click **Next**.



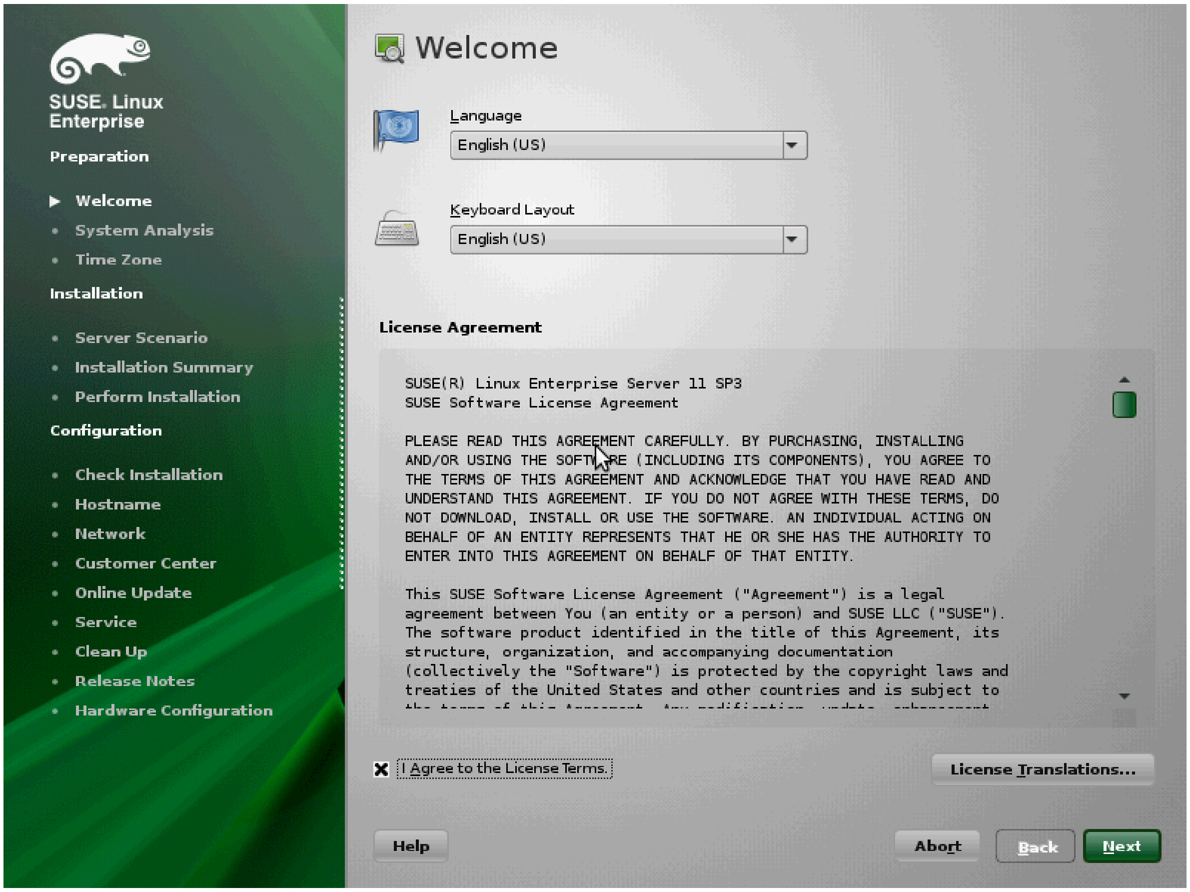
iii. Select the FC/FCoE LUN which was saved as boot device in system BIOS and click **Next**. Then proceed with the installation as usual.



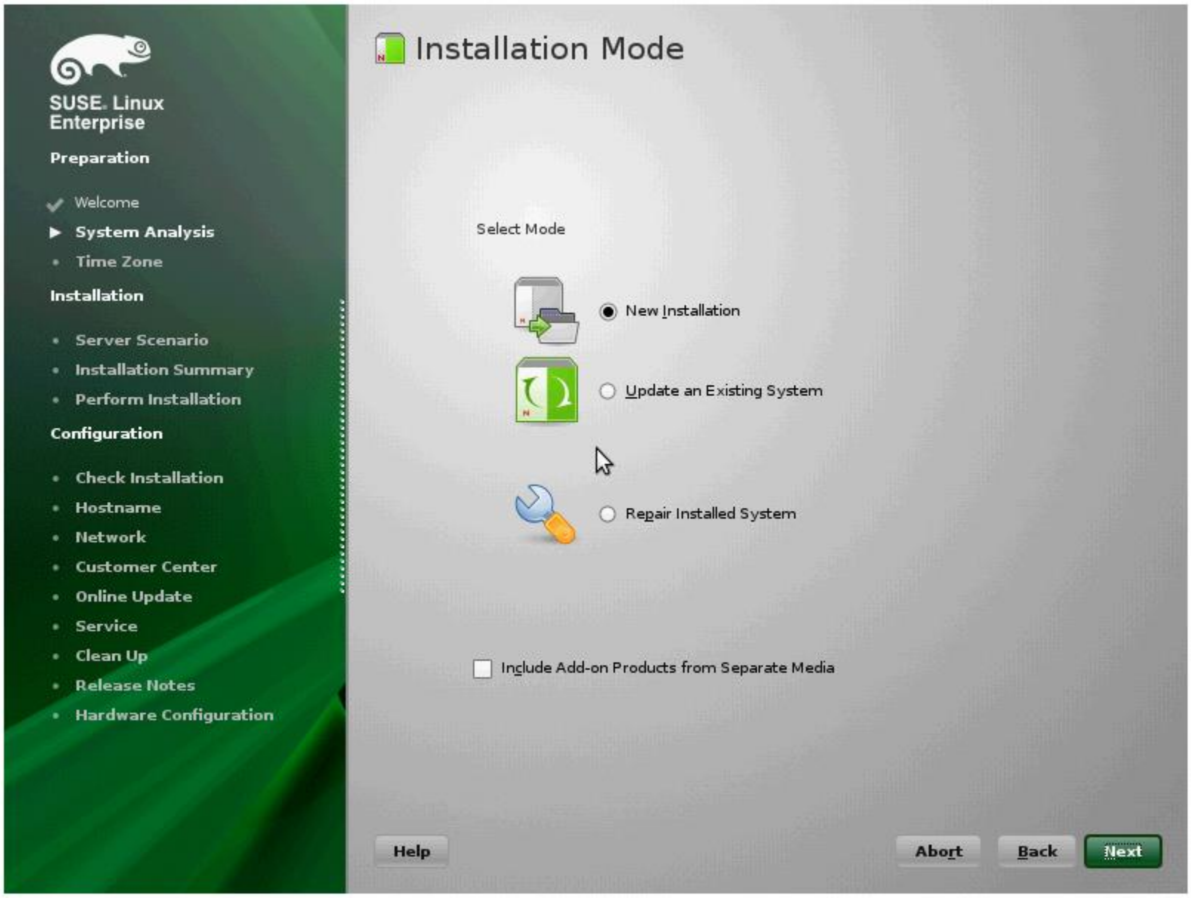
Note *Make sure the same LUN discovered at the Option ROM stage is selected for OS installation.*

8.2.3. SLES 11 SPx Installation

i. Choose installation language and Keyboard layout type. Select the checkbox **I Agree to the License terms** and click **Next**.

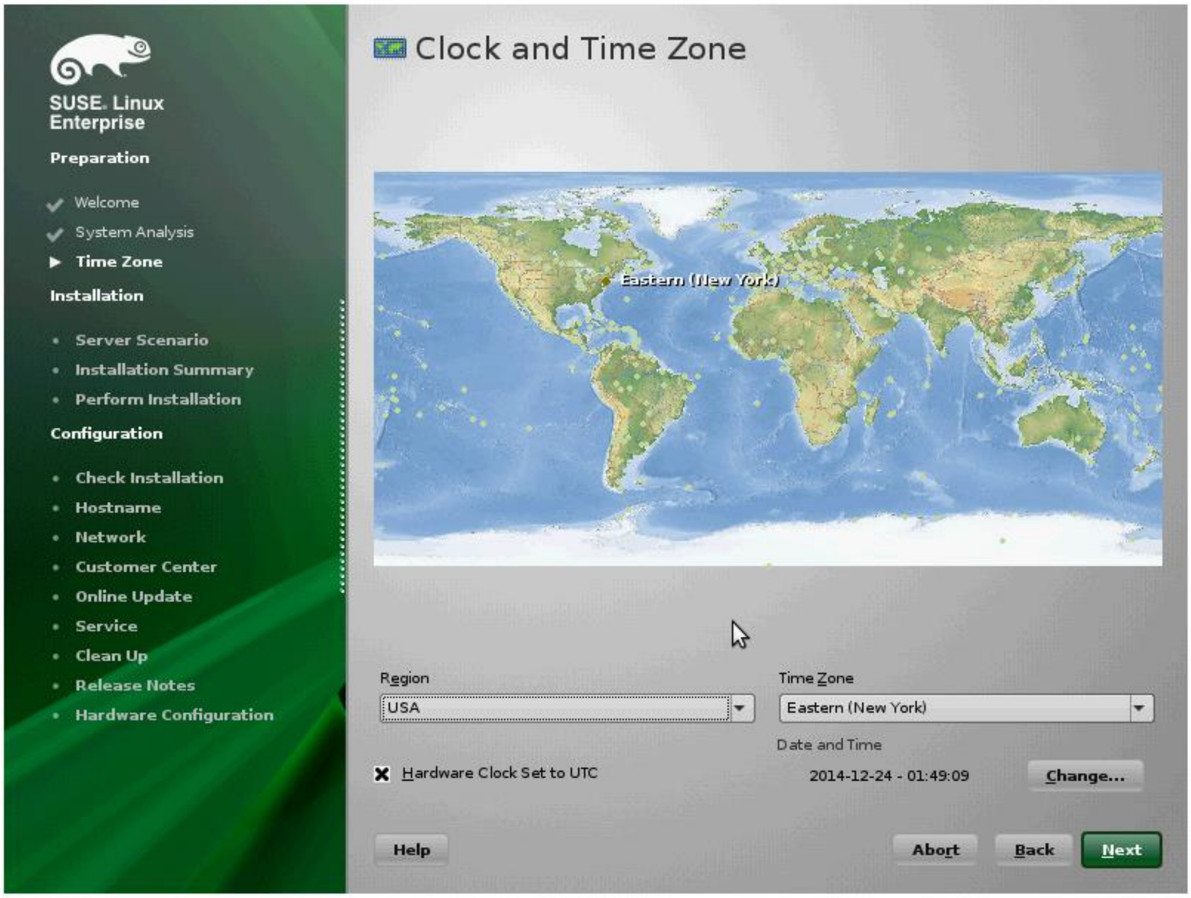


ii. Select **New Installation** to perform a fresh installation and click **Next**.



iii. Choose from the available base scenarios and click **Next**.

iv. Configure Clock and Time Zone settings. Click **Next**.

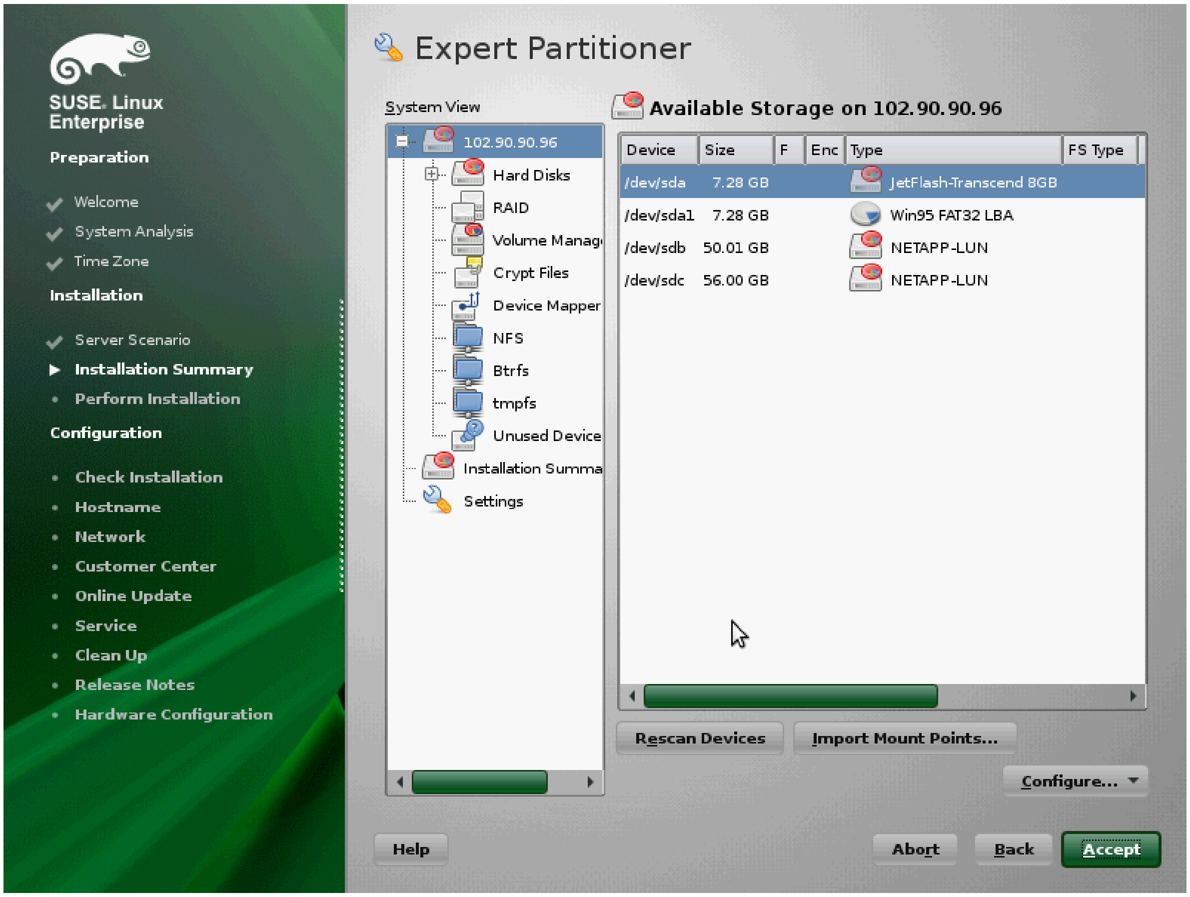


v. The discovered FC/FCoE LUNs will appear in the **Preparing Hard Disk** screen**.** Select the LUN which was saved as boot device in system BIOS. Click **Next**.

Note *Make sure the same LUN discovered at the Option ROM stage is selected for OS installation.*

vi. The **Installation Settings** screen displays the summary of user-selected and YaST-suggested options for the installation. You can review and modify them if required. Basic settings can be changed in the **Overview** tab and advanced settings can be changed in the **Expert** tab. To change, click on one of the headlines or click **Change** and select the category. Finally, click **Next**.

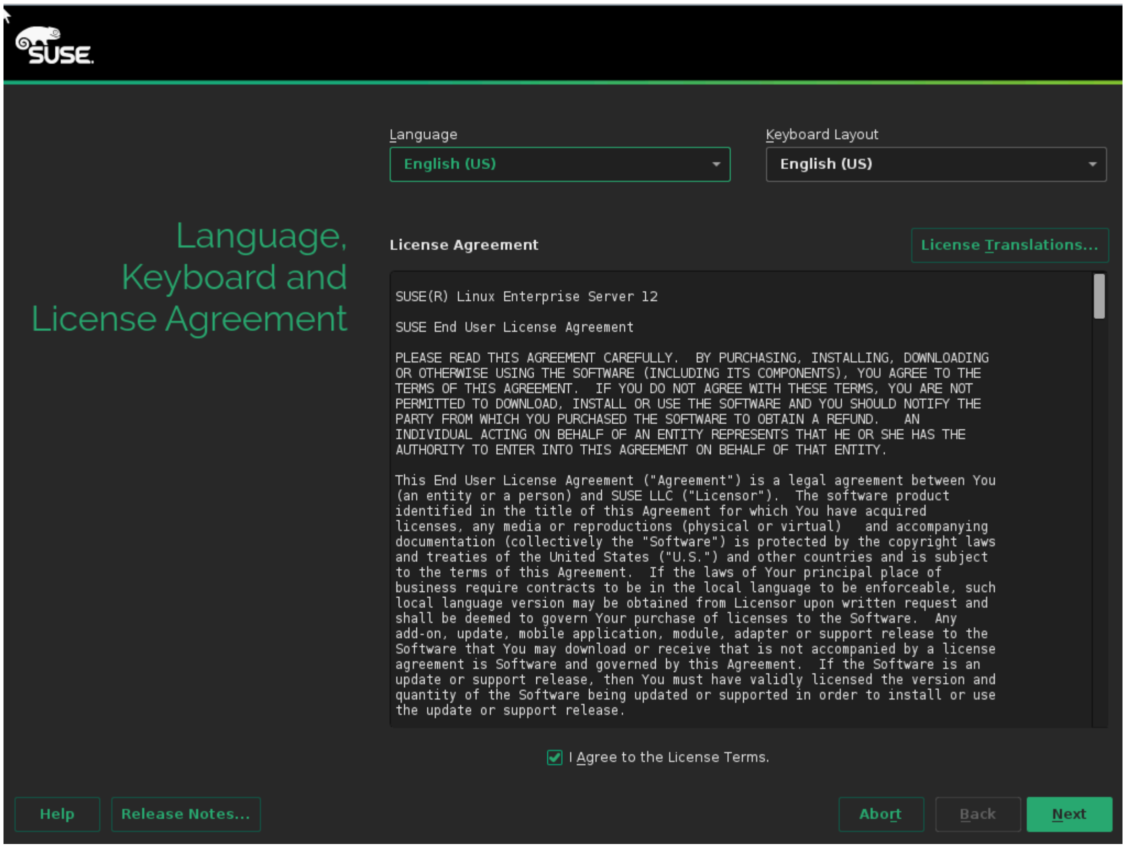
vii. The **Expert Partitioner** screen displays the partition setup suggested by the installer. Click on the device selected in step (v) and click **Accept**.



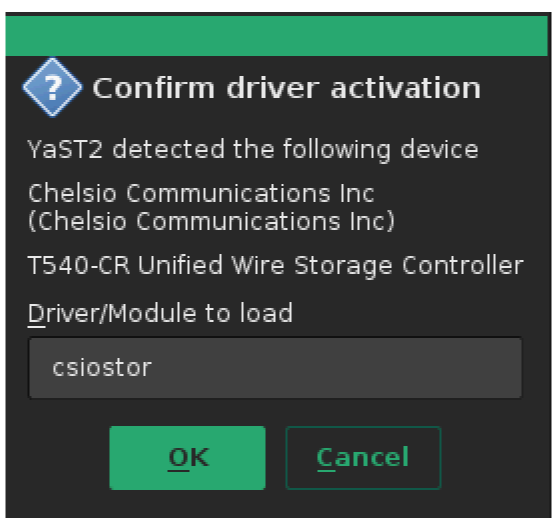
viii. Proceed with installation as usual

8.2.4. SLES 12/SLES 12 SPx Installation

i. Choose installation language and keyboard layout type. Select the checkbox **I Agree to the License terms** and click **Next**.

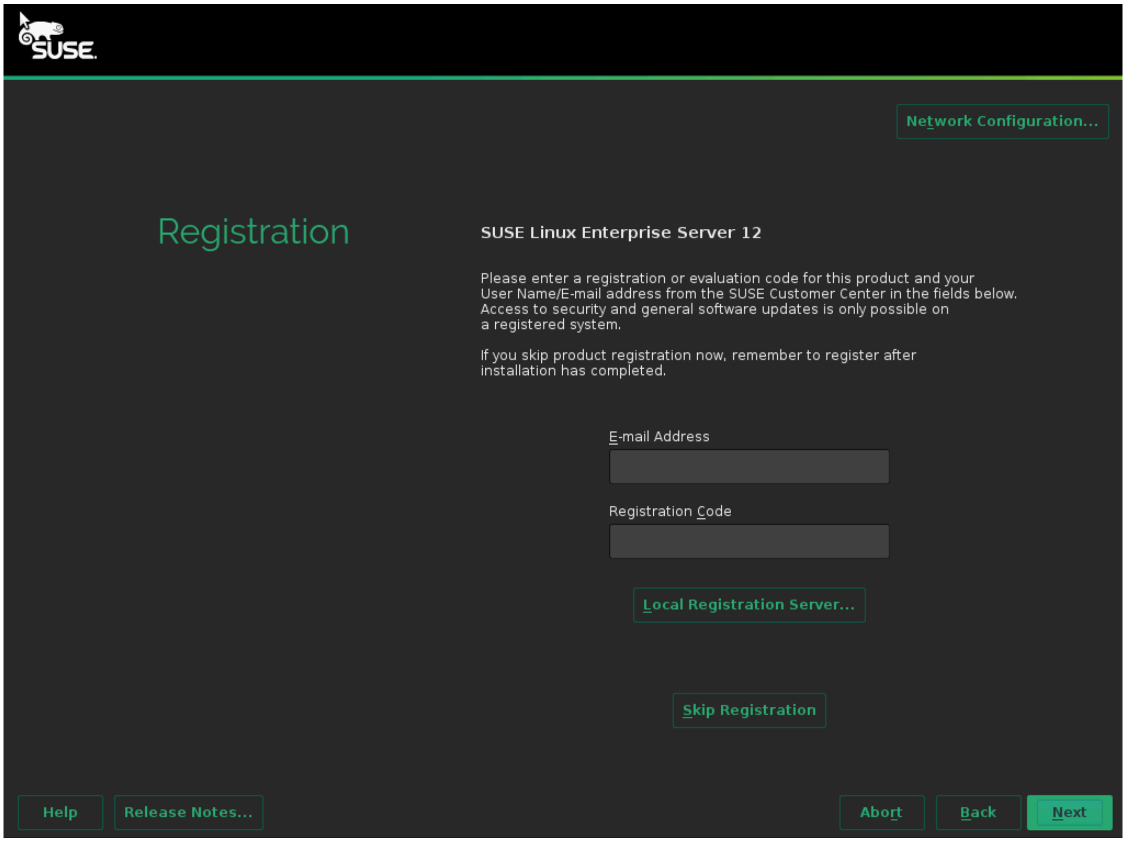


ii. During system probe, the YaST installer will detect Chelsio FCoE driver *csiostor* present in the DUD and prompt for confirmation to load/activate. Press [OK].

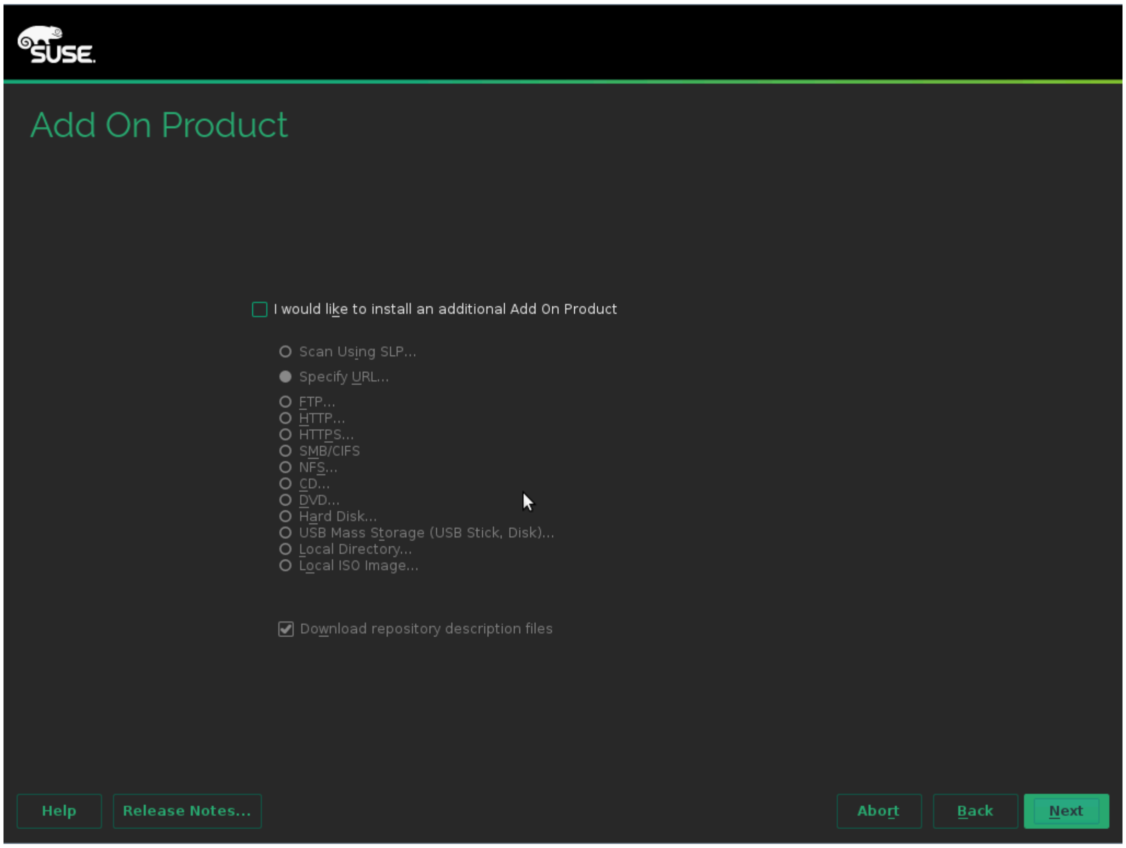


iii. To receive the latest updates for your operating system and technical support, you will need to register your system. Enter the registration or evaluation code for your copy of SLES 12 and email id associated with your Suse Customer Care (SCC) account and click **Next**.

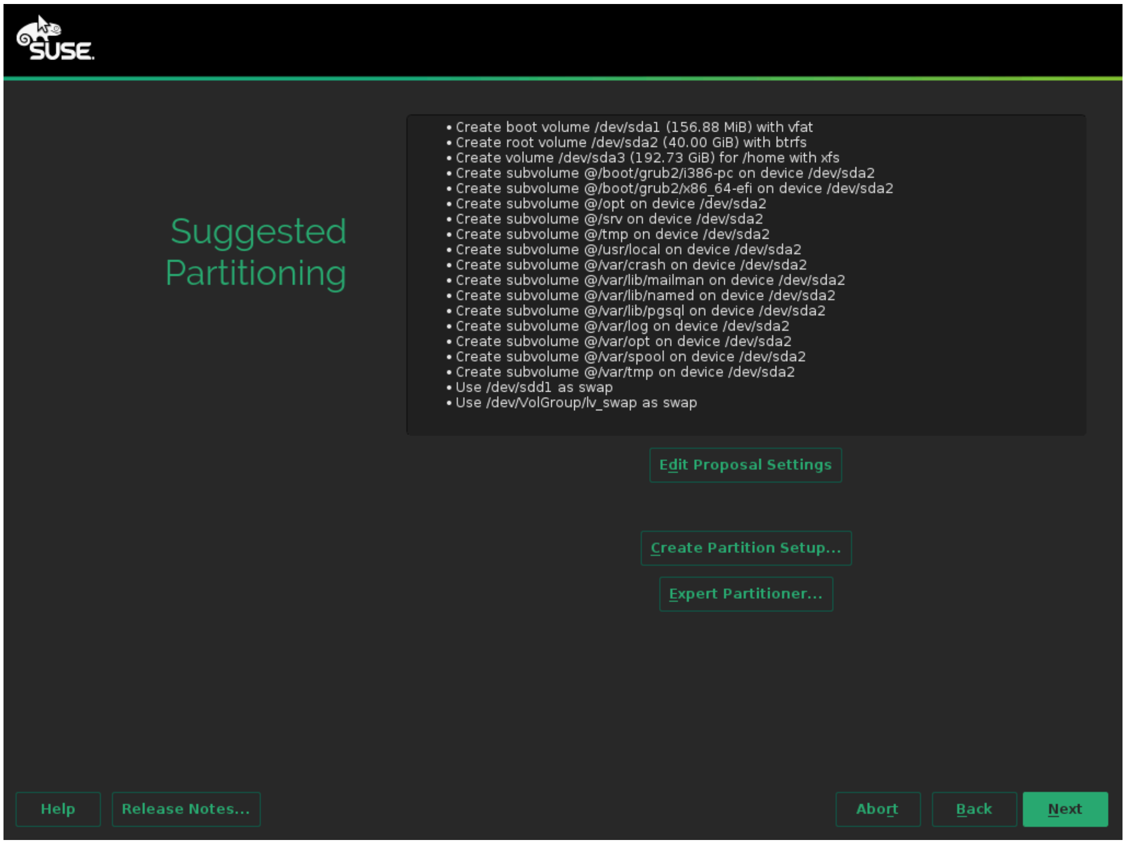
To bypass registration, click **Skip Registration** and then **Yes** on the pop-up window that appears.



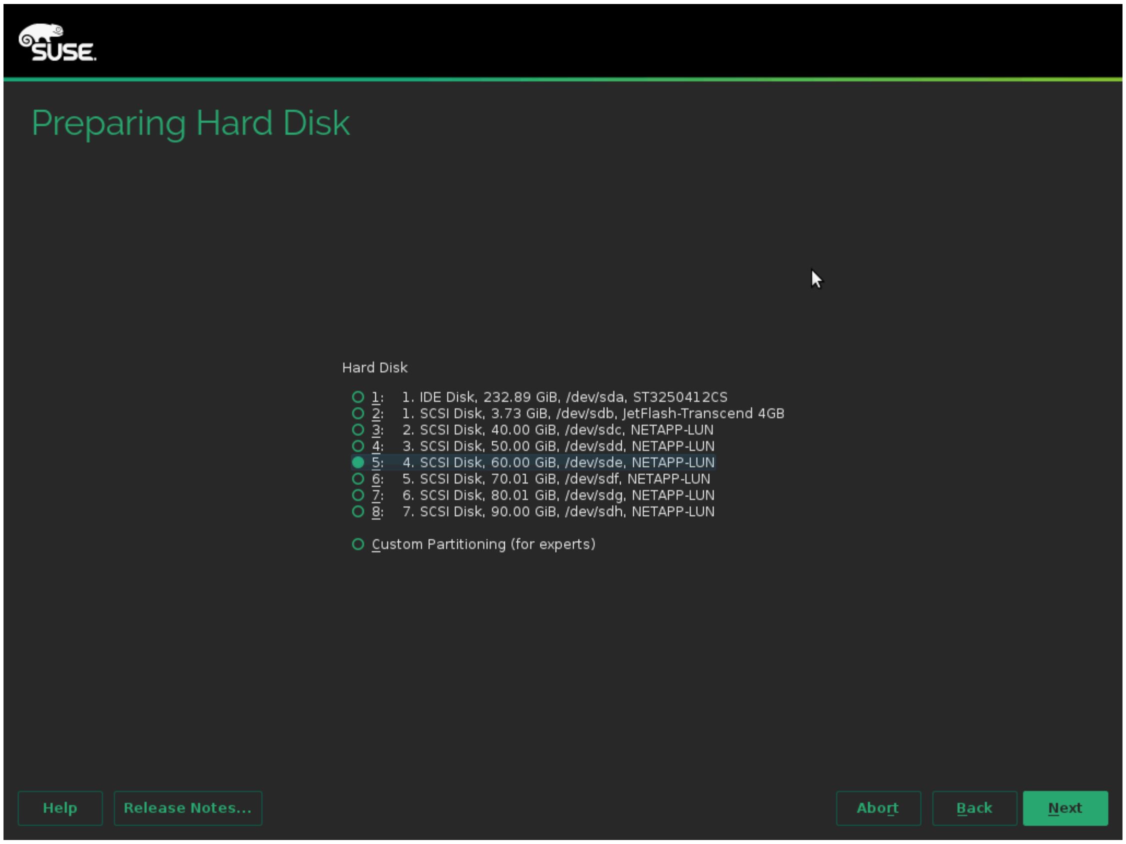
iv. The next screen will display a list of add-ons and extensions available for SLES 12. To install, select the checkbox *I would like to install an additional Add On Product*, then select the radio button for the add-on/extension you wish to install, and click **Next**.



v. On the *Suggested Partitioning* screen, YaST generated partition setup will be displayed. To change the suggested settings, click **Edit Proposal** Settings. To select the disk on which to apply the proposed settings, click **Create Partition Setup**. To change the partition setup click **Expert Partitioner**.

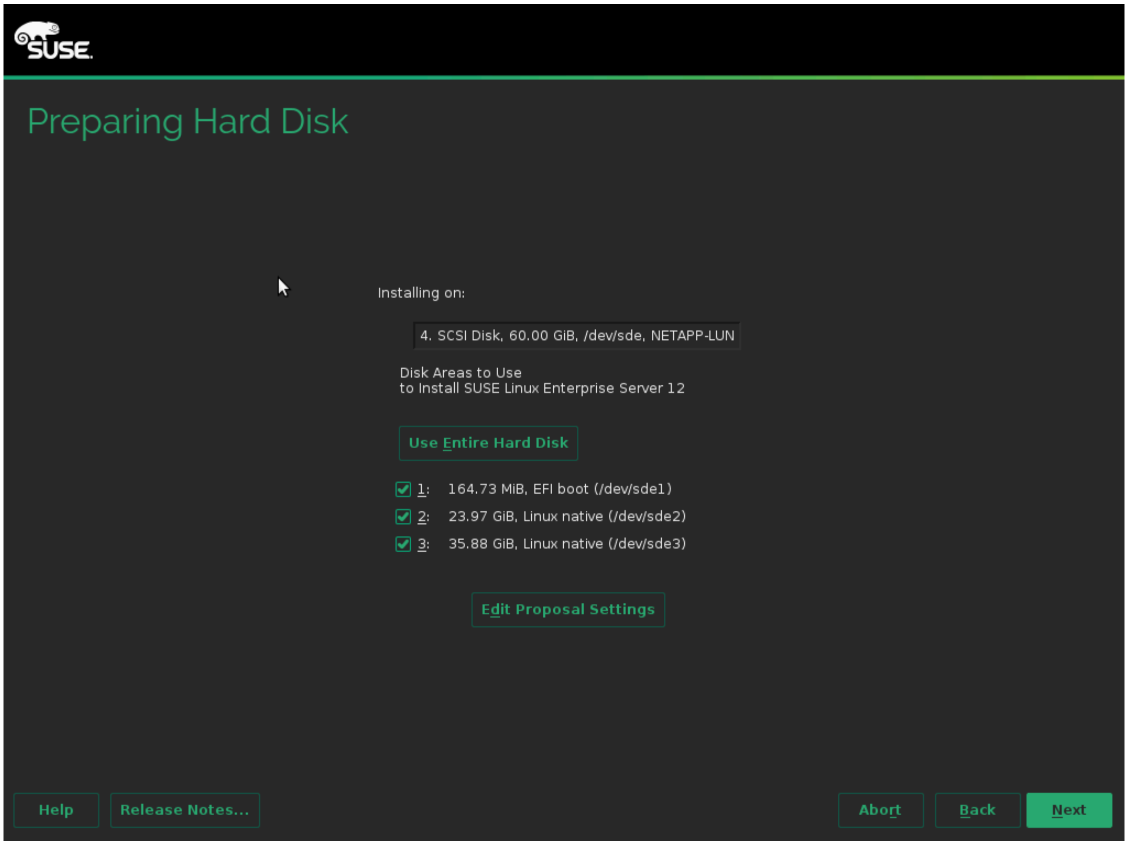


vi. Click **Create Partition Setup** and select the LUN which was saved as boot device in system BIOS. Click **Next**.

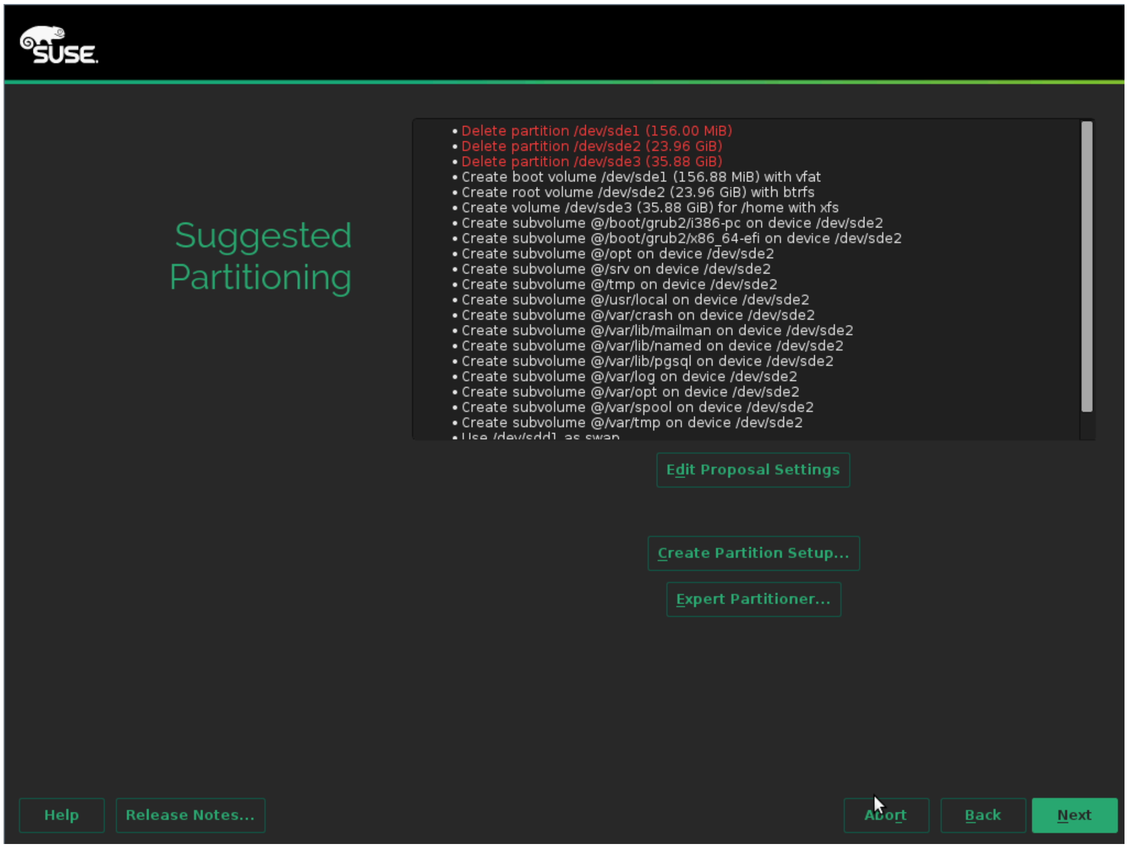


Note *Make sure the same LUN discovered at the Option ROM stage is selected for OS installation.*

vii. To use the entire LUN for installation, click **User Entire Hard Disk**. Please note that this will delete all the existing partitions. To install operating system on an existing partition, select the partition from the list displayed. Click **Next**.



viii. The selected LUN should now appear in the *Suggested Partitioning* screen*.* Click **Next**.



ix. Proceed with installation as usual.

8.3. Installation on iSCSI LUN

- If you are installing using CD/DVD, please make sure that the USB drive with DUD image is inserted. Also, change the boot priority to boot from CD/DVD in the BIOS setup.
    i. Insert the OS installation disc into your CD/DVD ROM.
    ii. On the Grub menu, choose *Install or upgrade an existing system* option if not already selected.
    iii. Type *e* and then *dd* at the boot prompt for RHEL 7. For RHEL 6 press *Tab* and then type *dd ip=ibft*. This will ensure that Chelsio iSCSI Initiator driver is used as SCSI transport medium. For SLES distributions, press *Tab* and then type *dd.*
    iv. Load Chelsio Driver Update Disk depending on the Linux distribution (**Click here** for RHEL 7.x; **Click here** for RHEL 6.x; **Click here** for SLES 11 SPx/SLES 12/SLES 12 SPx.

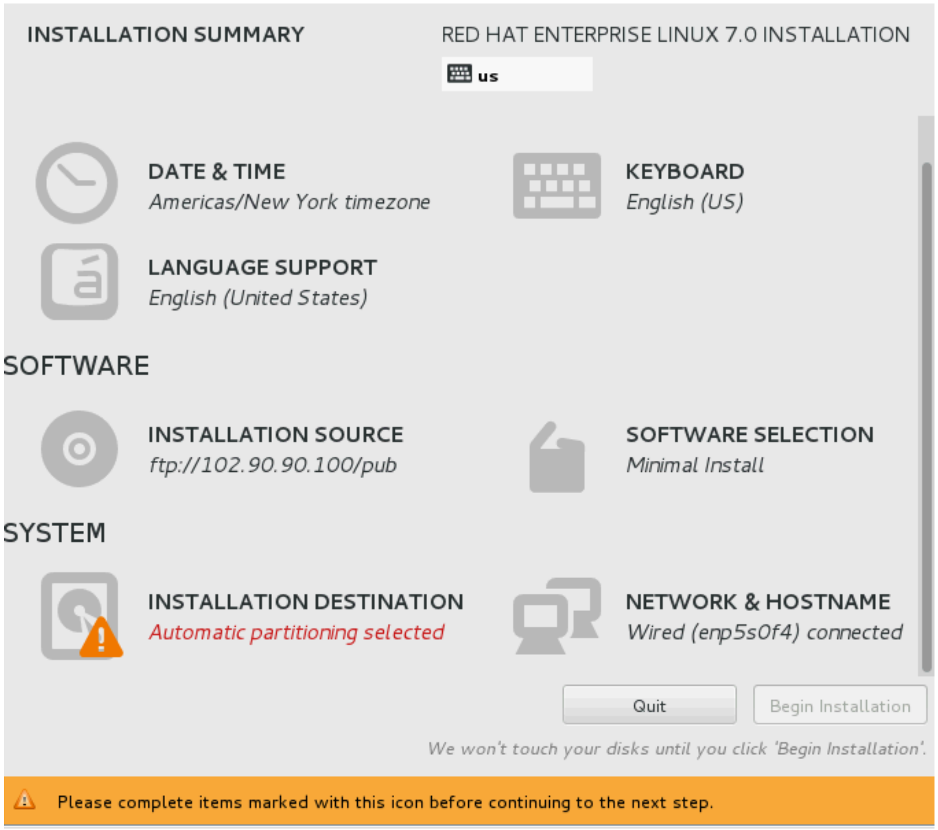
- If you are installing from a PXE server, please refer **8.1. Installation using Chelsio DUD**(**Click here** for RHEL 7.x; **Click here** for RHEL 6.x; **Click here** for SLES 11 SPx/SLES 12/SLES 12 SPx) section to load Chelsio Driver Update Disk.

After successfully loading Chelsio DUD, follow the procedure mentioned below to continue installation, based on the distribution.

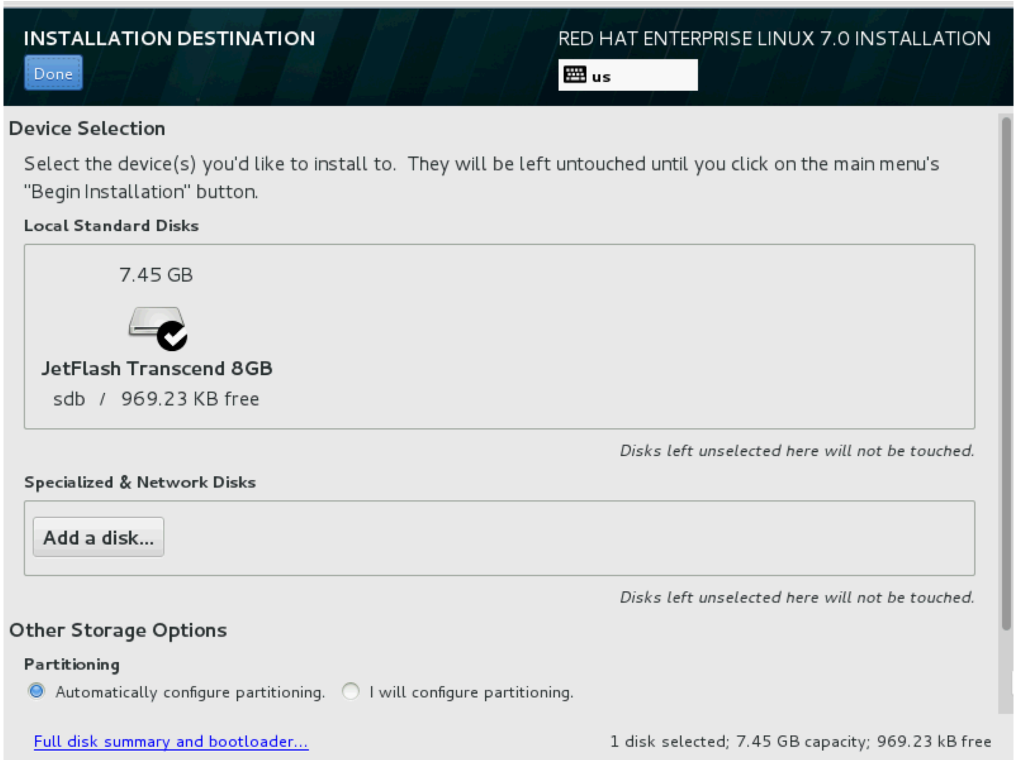
8.3.1. RHEL 7.x

i. On the installer welcome screen, choose your installation language and click **Continue**

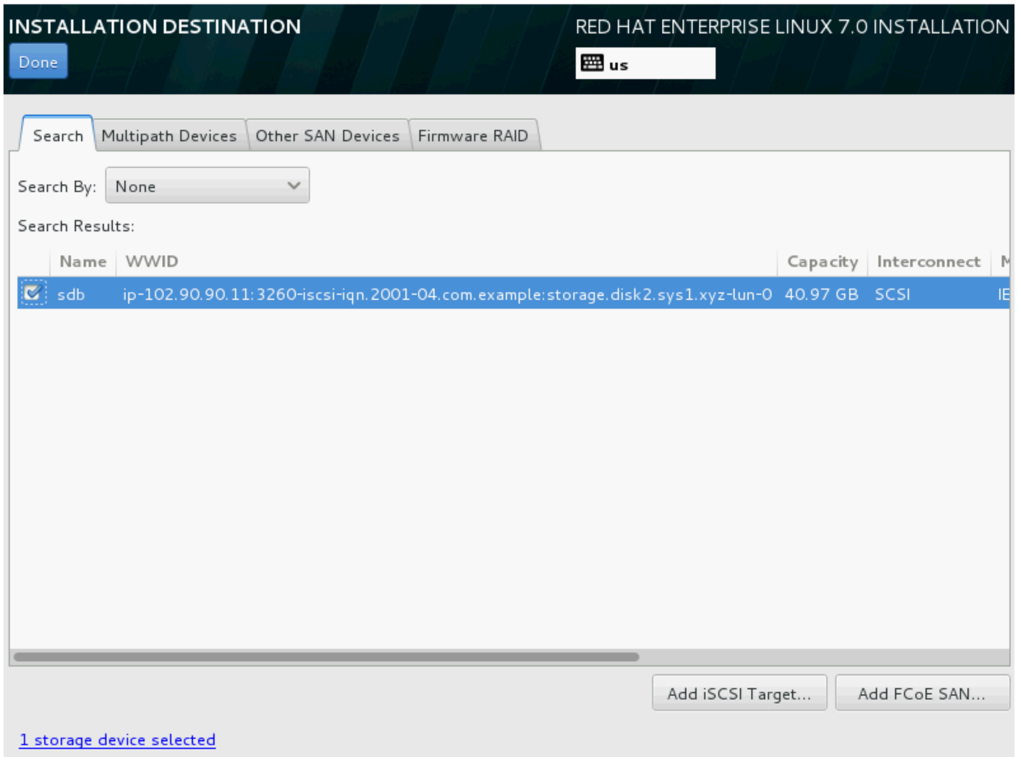
ii. Click **Installation Destination** under **SYSTEM.**



iii. Click **Add a disk**

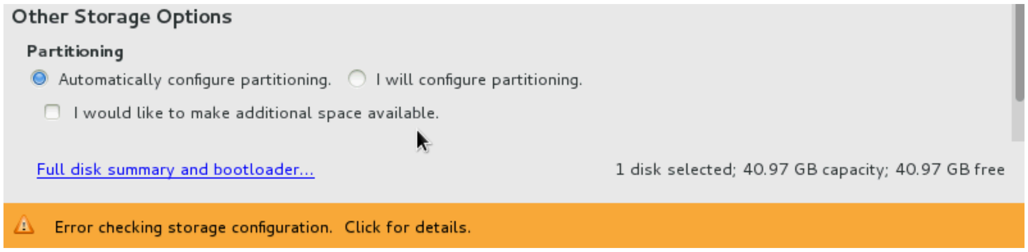


iv. The discovered iSCSI LUNs will appear in the **Search** tab. Select it and click **Done**.



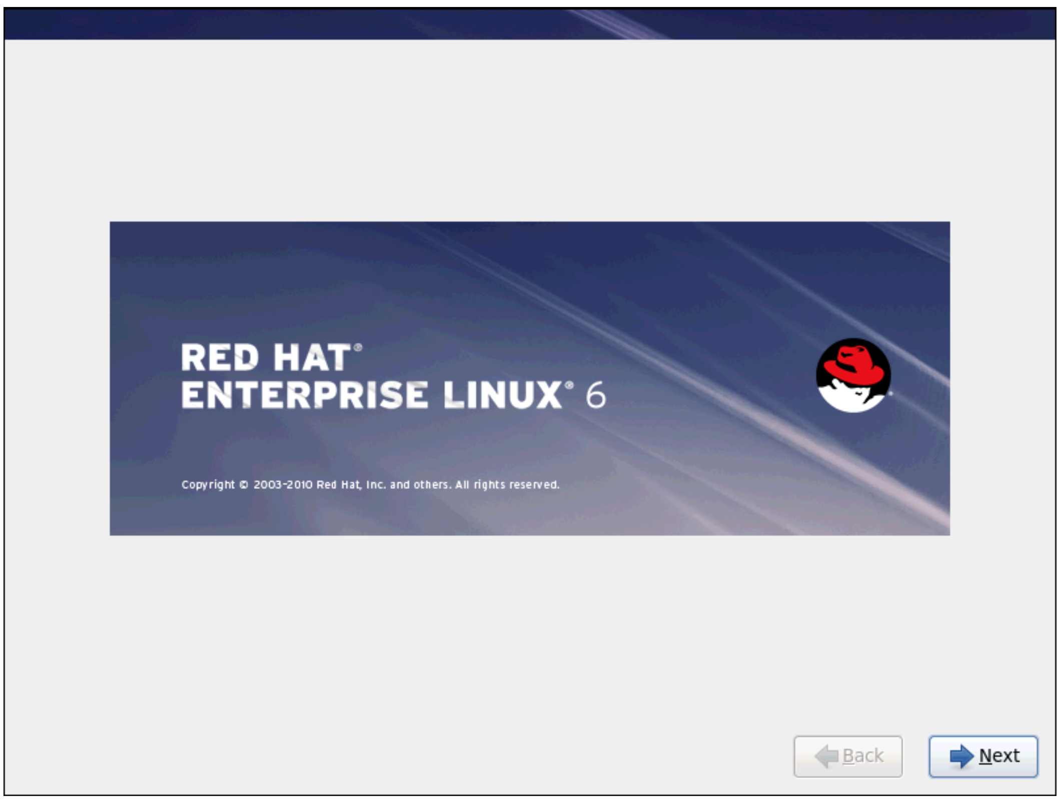
Note *Make sure the same LUN discovered at the Option ROM stage is selected for OS installation.*

v. Under **Other Storage Options**, you can either chose to configure partition automatically or manually. Select the appropriate option and click **Done**. Then proceed with the installation as usual.

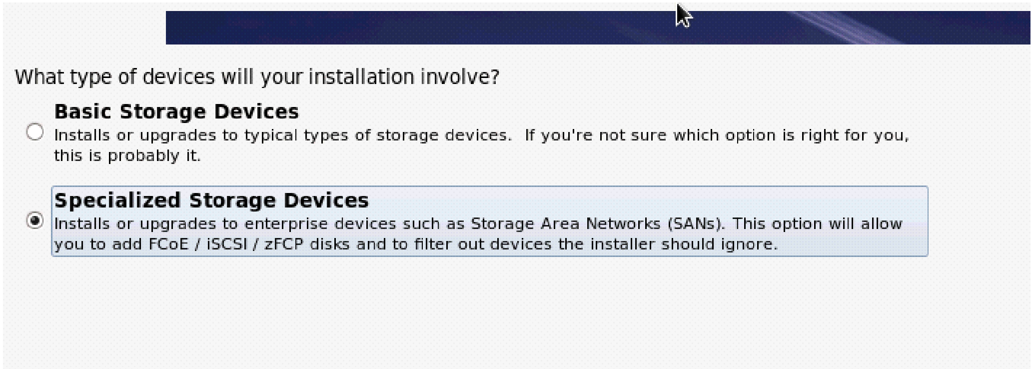


8.3.2. RHEL 6.x

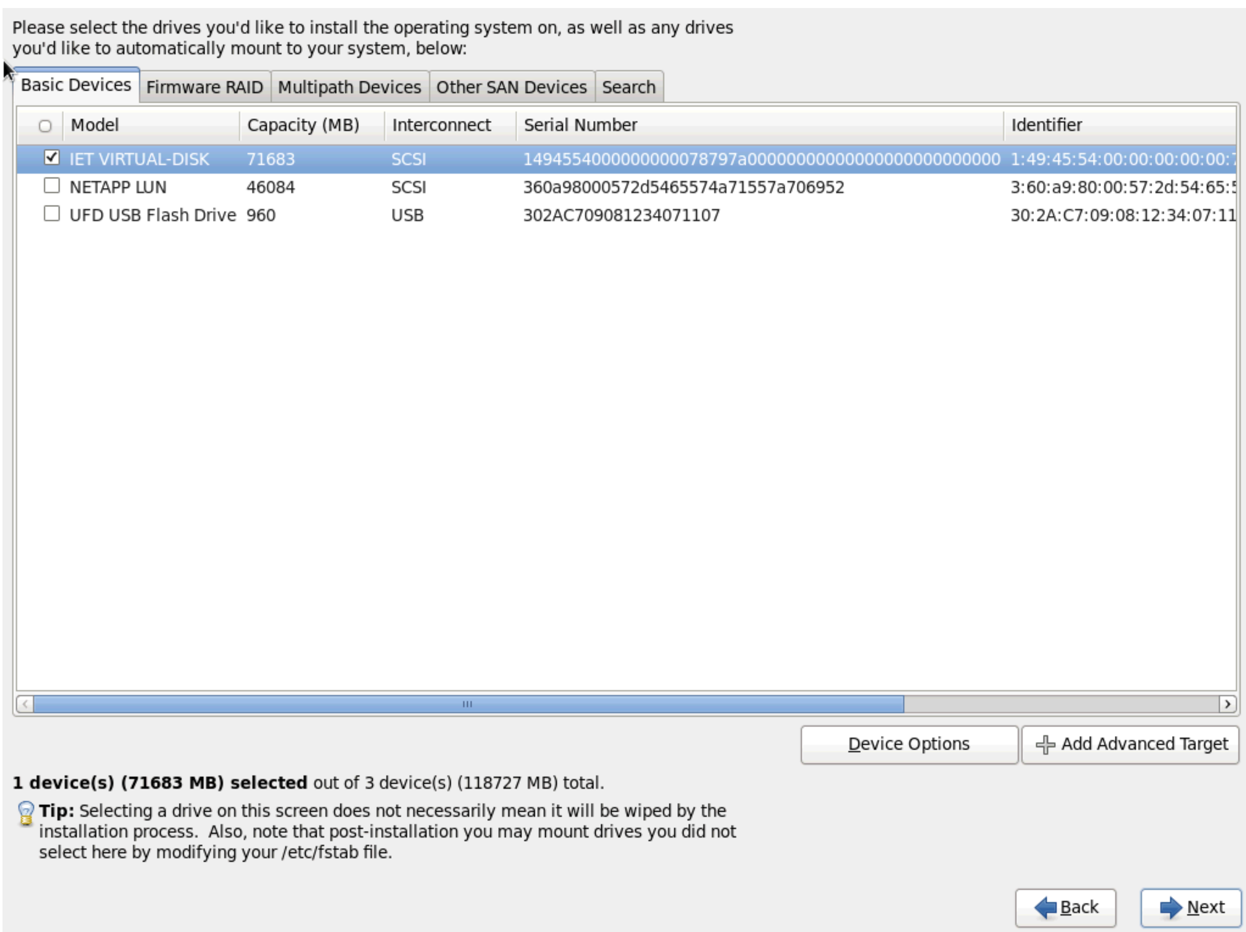
i. Click **Next** when the graphical installer screen appears.



ii. Select **Specialized Storage Devices** radio button and click **Next**.



iii. The discovered LUNs will appear in the **Basic Devices** tab. Select the LUN which was saved as boot device in system BIOS and click **Next**.

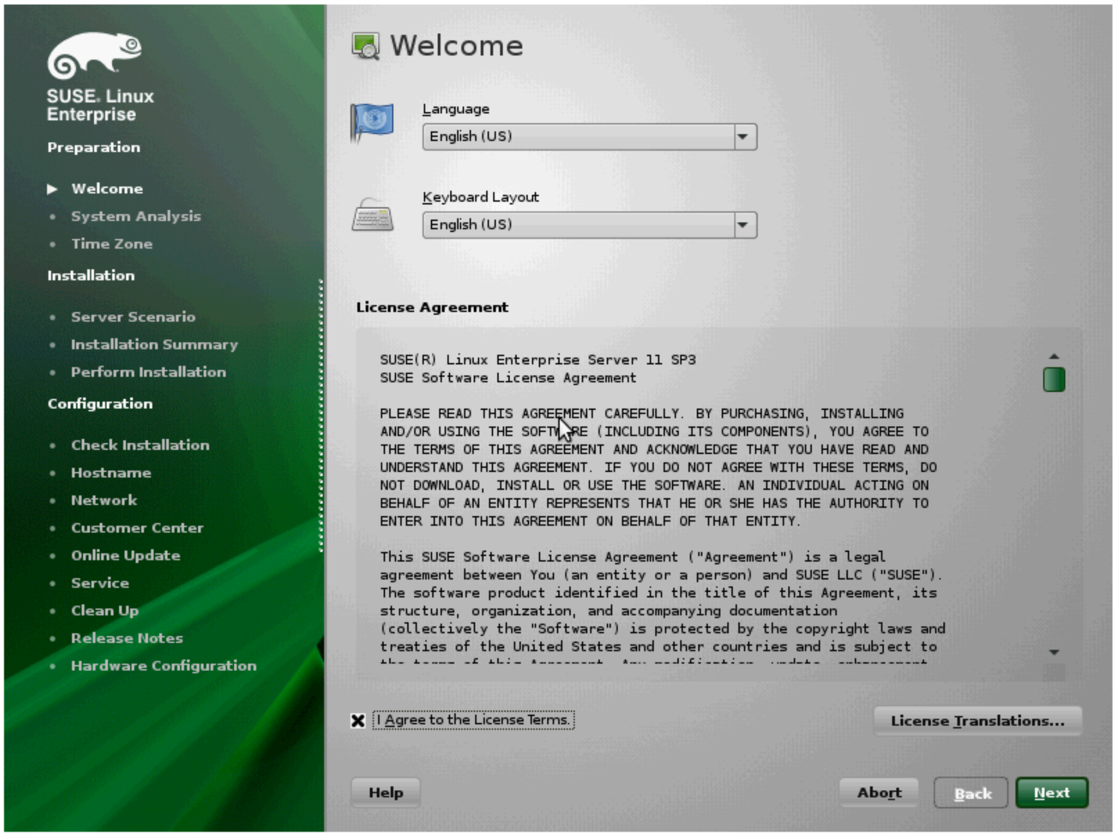


Note *Make sure the same LUN discovered at the Option ROM stage is selected for OS installation.*

iv. Proceed with the installation as usual.

8.3.3. SLES 11 SPx installation

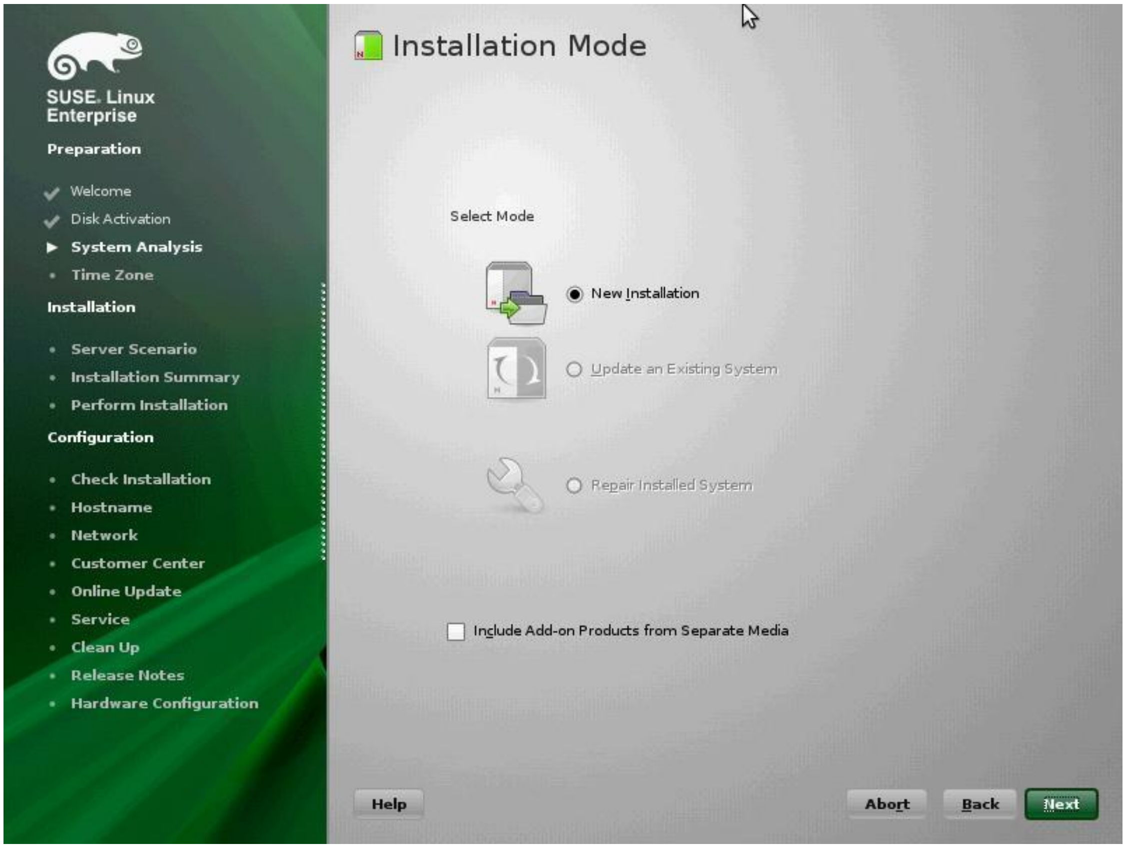
i. Choose installation language and Keyboard layout type. Select the checkbox **I Agree to the License terms** and click **Next**.



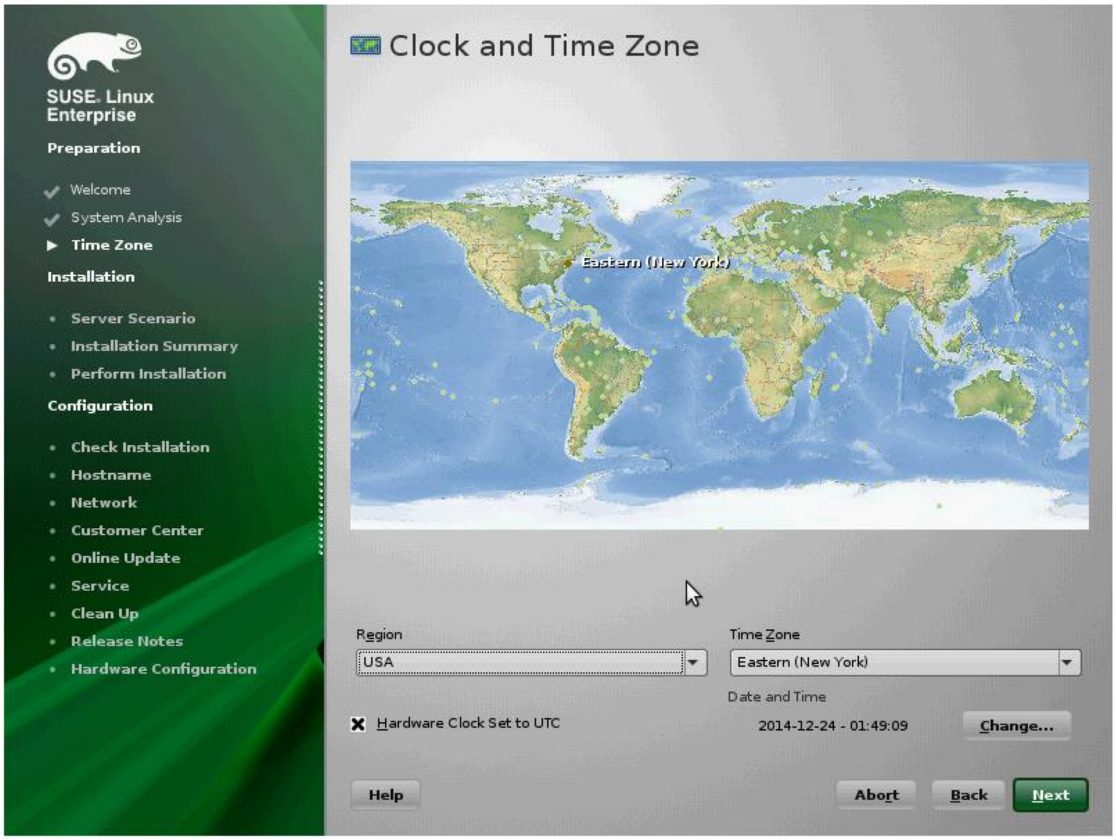
ii. Click **Configure iSCSI Disks** in the **Disk Activation** screen.

iii. The discovered LUNs will appear in the **Connected Targets** tab. Select the LUN which was saved as boot device in system BIOS and click **OK**.

iv. Select **New Installation** to perform a fresh installation and click **Next**.

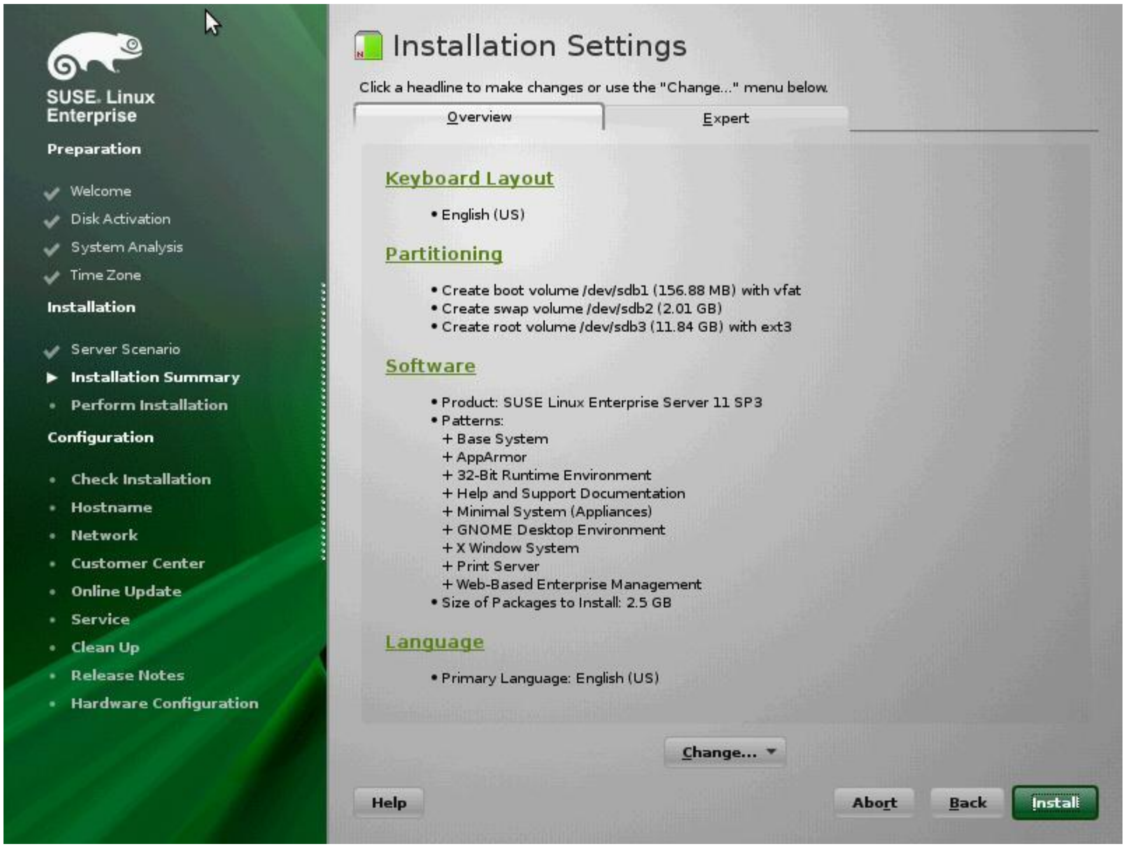


v. Configure Clock and Time Zone settings. Click **Next**.



vi. Choose from the available server base scenarios and click **Next**.

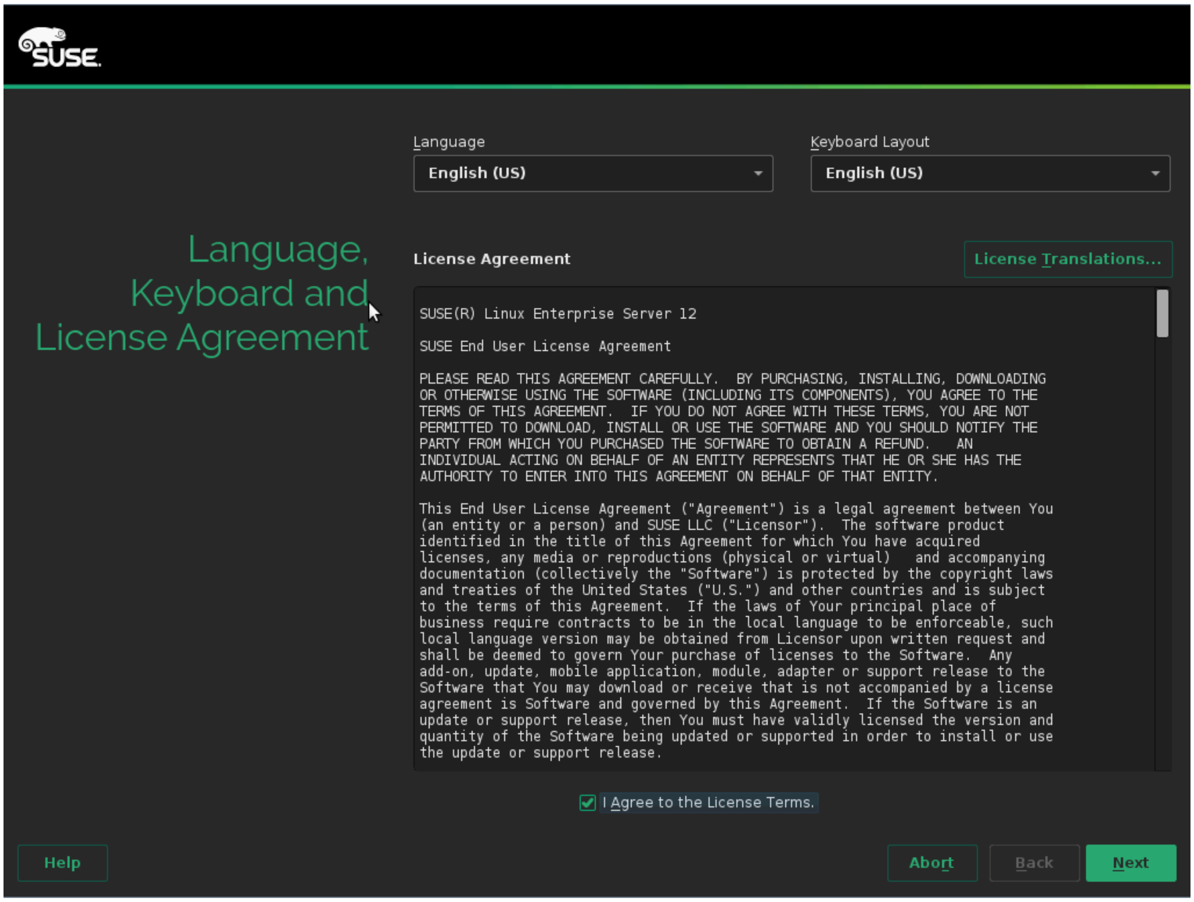
vii. The **Installation Settings** screen displays the summary of user-selected and YaST-suggested options for the installation. You can review and modify them if required. Basic settings can be changed in the **Overview** tab and advanced settings can be changed in the **Expert** tab. To change, click on one of the headlines or click **Change** and select the category. Finally, click **Next**.



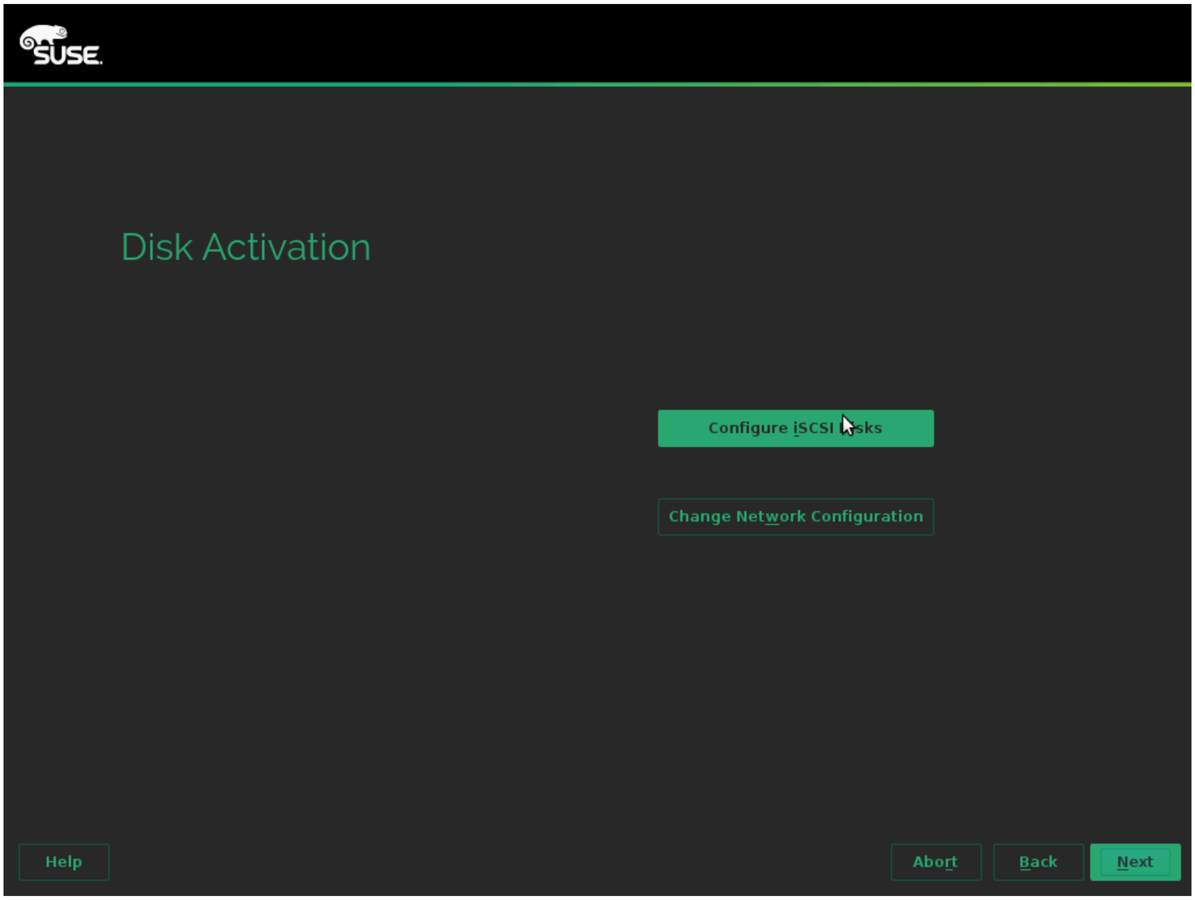
viii. Proceed with installation as usual.

8.3.4. SLES 12/SLES 12 SPx Installation

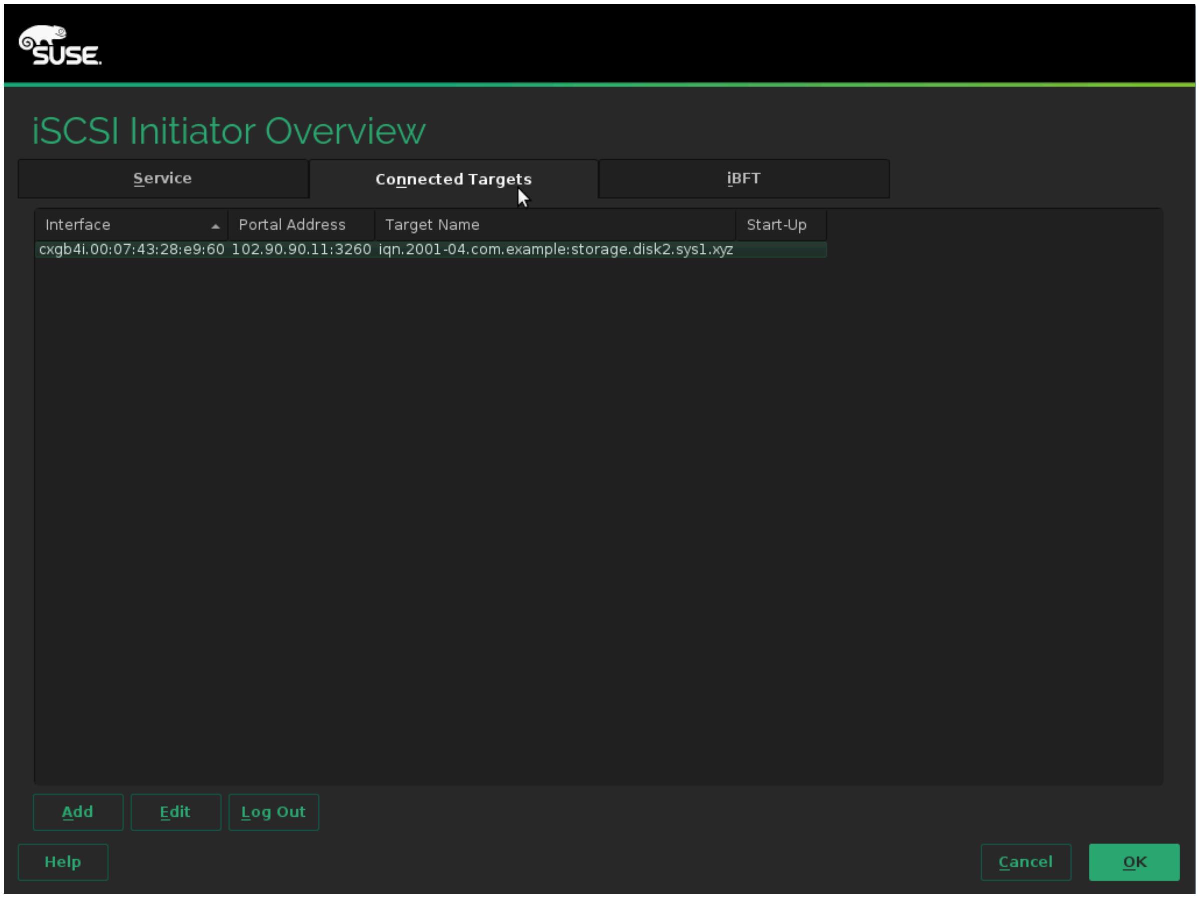
i. Choose installation language and Keyboard layout type. Select the checkbox **I Agree to the License terms** and click **Next**.



ii. Click **Configure iSCSI Disks** in the **Disk Activation** screen.



iii. The discovered LUNs will appear in the **Connected Targets** tab. Select the LUN which was saved as boot device in system BIOS and click **OK**.



Note *Make sure the same LUN discovered at the Option ROM stage is selected for OS installation.*

iv. Proceed with the installation as usual.

XXIII. Appendix A

1. Troubleshooting

- ***Cannot bring up Chelsio interface***

Make sure you have created the corresponding network-script configuration file as stated in **Cheslsio Unified Wire** chapter (See **Creating network-scripts**). If the file does exist, make sure the structure and contents are correct. A sample is given in the **Chelsio Unified Wire** chapter (See **Configuring network-scripts**). Another reason may be that the IP address mentioned in the configuration file is already in use on the network.

- ***Cannot ping through Chelsio interface***

First, make sure the interface was successfully brought up using ifup ethX (where ethX is your interface) and that it is linked to an IP address, either static or obtained through DHCP.

You then may want to check whether the destination host (i.e. the machine you are trying to ping) is up and running and accepts ICMP requests such as ping. If you get a return value of 0 when doing a `cat /proc/sys/net/ipv4/icmp_echo_ignore_all` on the remote host that means it is configured to reply to incoming pings. Change `ipv4` to `ipv6` in the path if you are using IPv6. Note that this is a Linux-only tip.

If you have more than one interface wired to the network, make sure you are using the right one for your outgoing ping requests. This can be done by using the `-I` option of the ping command, as shown in the following example:

```
[root@host~]# ping -I eth1 10.192.167.1
```

Where 10.192.167.1 is the machine you want to ping.

- ***Configuring firewall for your application***

In many cases the firewall software on the systems may prevent the applications from working properly. Please refer to the appropriate documentation for the Linux distribution on how to configure or disable the firewall.

- ***FCoE link not up***

Always enable LLDP on the interfaces as FCoE link won't come up until and unless a successful LLDP negotiation happens.

- ***priority-flow-control mode on the switch***

On the switch, make sure priority-flow-control mode is always set to auto and flow control is disabled.

- ***Configuring Ethernet interfaces on Cisco switch***

Always configure Ethernet interfaces on Cisco switch in trunk mode.

- ***Binding VFC to MAC***

If you are binding the VFC to MAC address in case of Cisco Nexus switch, then make sure you make the Ethernet interface part of both Ethernet VLAN and FCoE VLAN.

- **Cisco nexus switch reporting "pauseRateLimitErrDisable"**

If in any case the switch-port on the Cisco nexus switch is reporting "pauseRateLimitErrDisable", then perform an Ethernet port shut/no shut.

2. Chelsio End-User License Agreement (EULA)

Installation and use of the driver/software implies acceptance of the terms in the Chelsio End-User License Agreement (EULA).

IMPORTANT: PLEASE READ THIS SOFTWARE LICENSE CAREFULLY BEFORE DOWNLOADING OR OTHERWISE USING THE SOFTWARE OR ANY ASSOCIATED DOCUMENTATION OR OTHER MATERIALS (COLLECTIVELY, THE "SOFTWARE"). BY CLICKING ON THE "OK" OR "ACCEPT" BUTTON YOU AGREE TO BE BOUND BY THE TERMS OF THIS AGREEMENT. IF YOU DO NOT AGREE TO THE TERMS OF THIS AGREEMENT, CLICK THE "DO NOT ACCEPT" BUTTON TO TERMINATE THE INSTALLATION PROCESS.

1. License. Chelsio Communications, Inc. ("Chelsio") hereby grants you, the Licensee, and you hereby accept, a limited, non-exclusive, non-transferable license to install and use the Software with one or more Chelsio network adapters on a single server computer for use in communicating with one or more other computers over a network. You may also make one copy of the Software in machine readable form solely for back-up purposes, provided you reproduce Chelsio's copyright notice and any proprietary legends included with the Software or as otherwise required by Chelsio.

2. Restrictions. This license granted hereunder does not constitute a sale of the Software or any copy thereof. Except as expressly permitted under this Agreement, you may not:

(i) reproduce, modify, adapt, translate, rent, lease, loan, resell, distribute, or create derivative works of or based upon, the Software or any part thereof; or

(ii) make available the Software, or any portion thereof, in any form, on the Internet. The Software contains trade secrets and, in order to protect them, you may not decompile, reverse engineer, disassemble, or otherwise reduce the Software to a human-perceivable form. You assume full responsibility for the use of the Software and agree to use the Software legally and responsibly.

3. Ownership of Software. As Licensee, you own only the media upon which the Software is recorded or fixed, but Chelsio retains all right, title and interest in and to the Software and all subsequent copies of the Software, regardless of the form or media in or on which the Software may be embedded.

4. Confidentiality. You agree to maintain the Software in confidence and not to disclose the Software, or any information or materials related thereto, to any third party without the express written consent of Chelsio. You further agree to take all reasonable precautions to limit access of the Software only to those of your employees who reasonably require such access to perform their employment obligations and who are bound by confidentiality agreements with you.

5. Term. This license is effective in perpetuity, unless terminated earlier. You may terminate the license at any time by destroying the Software (including the related documentation), together with all copies or modifications in any form. Chelsio may terminate this license, and this license shall be deemed to have automatically terminated, if you fail to comply with any term or condition of this Agreement. Upon any termination, including termination by you, you must destroy the Software (including the related documentation), together with all copies or modifications in any form.

6. Limited Warranty. If Chelsio furnishes the Software to you on media, Chelsio warrants only that the media upon which the Software is furnished will be free from defects in material or workmanship under normal use and service for a period of thirty (30) days from the date of delivery to you.

CHELSIO DOES NOT AND CANNOT WARRANT THE PERFORMANCE OR RESULTS YOU MAY OBTAIN BY USING THE SOFTWARE OR ANY PART THEREOF. EXCEPT FOR THE FOREGOING LIMITED WARRANTY, CHELSIO MAKES NO OTHER WARRANTIES, EXPRESS OR IMPLIED, AND HEREBY DISCLAIMS ALL OTHER WARRANTIES, INCLUDING, BUT NOT LIMITED TO, NON-INFRINGEMENT OF THIRD PARTY RIGHTS, MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow the exclusion of implied warranties or limitations on how long an implied warranty may last, so the above limitations may not apply to you. This warranty gives you specific legal rights and you may also have other rights which vary from state to state.

7. Remedy for Breach of Warranty. The sole and exclusive liability of Chelsio and its distributors, and your sole and exclusive remedy, for a breach of the above warranty, shall be the replacement of any media furnished by Chelsio not meeting the above limited warranty and which is returned to Chelsio. If Chelsio or its distributor is unable to deliver replacement media which is free from defects in materials or workmanship, you may terminate this Agreement by returning the Software.

8. Limitation of Liability. IN NO EVENT SHALL CHELSIO HAVE ANY LIABILITY TO YOU OR ANY THIRD PARTY FOR ANY INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL OR PUNITIVE DAMAGES, HOWEVER CAUSED, AND ON ANY THEORY OF LIABILITY, ARISING OUT OF OR RELATED TO THE LICENSE OR USE OF THE SOFTWARE, INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR LOSS OF ANTICIPATED PROFITS, EVEN IF CHELSIO HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN NO EVENT SHALL CHELSIO'S LIABILITY ARISING OUT OF OR RELATED TO THE LICENSE OR USE OF THE SOFTWARE EXCEED THE AMOUNTS PAID BY YOU FOR THE LICENSE GRANTED HEREUNDER. THESE LIMITATIONS SHALL APPLY NOTWITHSTANDING ANY FAILURE OF ESSENTIAL PURPOSE OF ANY LIMITED REMEDY.

9. High Risk Activities. The Software is not fault-tolerant and is not designed, manufactured or intended for use or resale as online equipment control equipment in hazardous environments requiring fail-safe performance, such as in the operation of nuclear facilities, aircraft navigation or communication systems, air traffic control, direct life support machines, or weapons systems, in which the failure of the Software could lead directly to death, personal injury, or severe physical or environmental damage. Chelsio specifically disclaims any express or implied warranty of fitness for any high risk uses listed above.

10. Export. You acknowledge that the Software is of U.S. origin and subject to U.S. export jurisdiction. You acknowledge that the laws and regulations of the United States and other countries may restrict the export and re-export of the Software. You agree that you will not export or re-export the Software or documentation in any form in violation of applicable United States and foreign law. You agree to comply with all applicable international and national laws that apply to the Software, including the U.S.

Export Administration Regulations, as well as end-user, end-use, and destination restrictions issued by U.S. and other governments.

11. Government Restricted Rights. The Software is subject to restricted rights as follows. If the Software is acquired under the terms of a GSA contract: use, reproduction or disclosure is subject to the restrictions set forth in the applicable ADP Schedule contract. If the Software is acquired under the terms of a DoD or civilian agency contract, use, duplication or disclosure by the Government is subject

to the restrictions of this Agreement in accordance with 48 C.F.R. 12.212 of the Federal Acquisition Regulations and its successors and 49 C.F.R. 227.7202-1 of the DoD FAR Supplement and its successors.

12. General. You acknowledge that you have read this Agreement, understand it, and that by using the Software you agree to be bound by its terms and conditions. You further agree that it is the complete and exclusive statement of the agreement between Chelsio and you, and supersedes any proposal or prior agreement, oral or written, and any other communication between Chelsio and you relating to the subject matter of this Agreement. No additional or any different terms will be enforceable against Chelsio unless Chelsio gives its express consent, including an express waiver of the terms of this Agreement, in writing signed by an officer of Chelsio. This Agreement shall be governed by California law, except as to copyright matters, which are covered by Federal law. You hereby irrevocably submit to the personal jurisdiction of, and irrevocably waive objection to the laying of venue (including a waiver of any argument of forum non conveniens or other principles of like effect) in, the state and federal courts located in Santa Clara County, California, for the purposes of any litigation undertaken in connection with this Agreement. Should any provision of this Agreement be declared unenforceable in any jurisdiction, then such provision shall be deemed severable from this Agreement and shall not affect the remainder hereof. All rights in the Software not specifically granted in this Agreement are reserved by Chelsio. You may not assign or transfer this Agreement (by merger, operation of law or in any other manner) without the prior written consent of Chelsio and any attempt to do so without such consent shall be void and shall constitute a material breach of this Agreement.

Should you have any questions concerning this Agreement, you may contact Chelsio by writing to:

Chelsio Communications, Inc.
209 North Fair Oaks Avenue,
Sunnyvale, CA 94085
U.S.A