Using HTTP Boot to Install an Operating System on Lenovo ThinkSystem servers

Last Update: May 2019

Introduces the use of HTTP Boot as a way to deploy servers over the network

Compares HTTP Boot with the existing PXE Boot functionality

Demonstrates the usage of HTTP Boot on Lenovo ThinkSystem servers

Provides instructions for use with SUSE Linux Enterprise Server 12

Neo Cui

Abstract

HTTP Boot is client-server communication based application. It combines the Dynamic Host Configuration Protocol (DHCP), Domain Name System (DNS), and Hypertext Transfer Protocol (HTTP) to provide the capability for system deployment and configuration over the network. This new capability can be utilized as a higher-performance replacement for Trivial File Transfer Protocol (TFTP) based Preboot Execution Environment (PXE) Boot methods of network deployment.

This document provides a brief introduction to the HTTP Boot mechanism, instructions on setting up the HTTP Boot server on SUSE Linux Enterprise Server 12 SP2, and a practice of installing SLES 12 SP2 on a Lenovo® ThinkSystem™ server using HTTP Boot. HTTP Boot is also supported on SLES 12 SP3, SLES 12 SP4 and SLES 15.

This paper is intended for IT administrators. Readers are expected to have the basic knowledge of network deployment.

Contents

Introduction

UEFI Specification V2.5 introduces new protocols that are related to the HTTP Boot in network stack. HTTP Boot is one method of booting a server from a Uniform Resource Identifier (URI), using HTTP technology. Users can boot a Network Boot Program (NBP) with HTTP Boot technology.

Compared to PXE Boot, HTTP Boot can handle much larger files than TFTP, and scale to much larger distances. You can easily download multi-megabyte files, such as a Linux kernel and a root file system, and you can download from servers that are not on your local area network.

HTTP Boot is recommended if you would like to deploy OS in a quicker and more stable way. This section introduces the typical deployment scenario of HTTP Boot and explains the differences between PXE Boot and HTTP Boot.

SLES support of HTTP Boot on ThinkSystem servers:

- SLES 12 SP2, SLES 12 SP3, SLES 12 SP4 and SLES 15 support HTTP Boot.
- ThinkSystem UEFI code released November 2018 dropped support of HTTP Boot on SLES 12 SP2.
- For SLES 12 SP3, upgrade grub2 per these guidelines:

  `https://www.suse.com/support/update/announcement/2018/suse-ru-20183378-1/`

Deployment scenario

A typical network configuration supporting UEFI HTTP Boot involves several network-related systems, either HTTP, DHCP and DNS servers deployed on one physical machine, or separate physical machines. The HTTP Boot client(s) can access the HTTP Boot server via the HTTP protocol in the system. For more information about HTTP Boot, refer to:

`https://github.com/tianocore/tianocore.github.io/wiki/HTTP-Boot`

Figure 1 shows a simple HTTP Boot network topology.

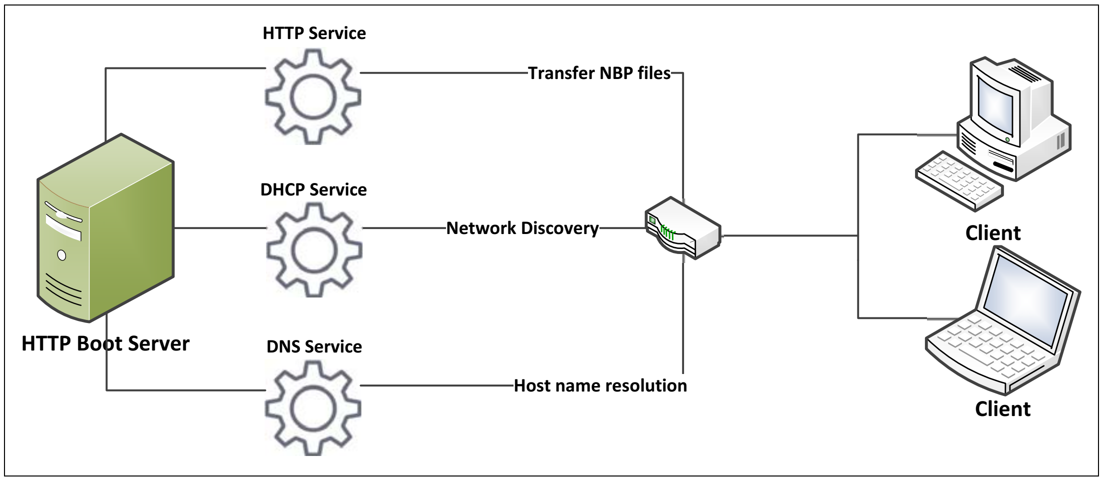


Figure 1 HTTP Boot network topology

The key components in Figure 1 are as follows:

- **HTTP Boot Server** is a server with HTTP service, DHCP service, and DNS service enabled to perform the HTTP Boot function.
- **HTTP service** could be located either inside the local intranet or across networks, such as on the Internet. The Network Boot Program (NBP) is deployed on the HTTP server. In this example, The NBP file called "core.efi" is used as the boot resource. NBP is executed on the HTTP Boot client, which may include installing an operating system, or running a service OS for maintenance and recovery tasks.
- **DHCP service** with HTTP Boot extension works for boot service discovery. Besides the standard host configuration information (such as address, subnet, name-server), the DHCP service can also provide the NBP locations in URI format on the HTTP server.
- **DNS service** is optional for the HTTP Boot system. It provides standard domain name resolution service if you want to translate more readily memorized domain names to the numerical IP addresses.
- **Clients** (UEFI HTTP Boot clients) are the PCs or servers that initiate the communication to HTTP Boot server where the OS will ultimately be installed.

Differences between PXE Boot and HTTP Boot

HTTP Boot is a function in the UEFI specification that replaces and improves on the functions provided by PXE Boot.

The PXE Boot protocol layout is shown in Figure 2. PXE makes the Network Interface Card (NIC) function like a boot device. The PXE-enabled NIC on the client sends out a broadcast request to the DHCP server, which returns with an IP address for the client along with the address of the TFTP server, as well as the location of boot files on the TFTP server.

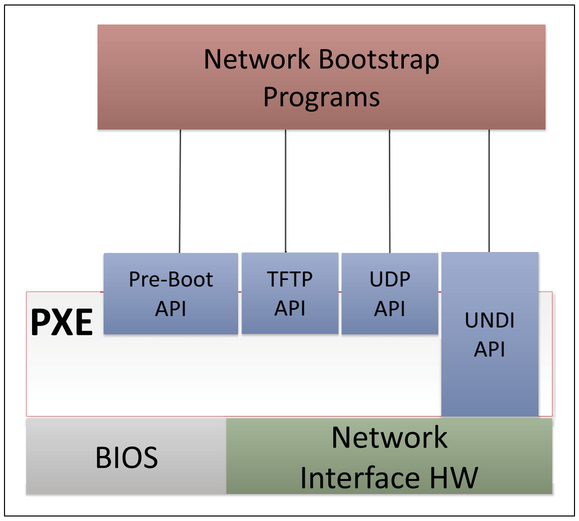


Figure 2 PXE Boot architecture

Over time, as network technology improved the shortcomings of PXE Boot, became more and more apparent:

- Security
  - No encryption or authentication mechanism
  - Allows man-in-the-middle attacks on the DHCP server
- Scaling issue

  - TFTP time-outs / UDP packet loss
  - Low data transmission speed

HTTP Boot technology is introduced in UEFI specification 2.5 to enhance network boot function, much as PXE Boot does. However, HTTP Boot has the following advantages:

- HTTPS security mechanism based on Transport Layer Security (TLS)
- TCP reliability
- HTTP load balancing

HTTP Boot architecture

This section introduces the HTTP Boot architecture.

HTTP Boot protocol

Figure 3 illustrates the UEFI network layers related to HTTP Boot workflow. The HTTP Boot driver is layered on the top of the UEFI network stack implementation. The UEFI network stack implementation uses DHCP service for boot service discovery, and uses DNS service for domain name resolution if needed. It also consumes HTTP service to retrieve boot files from the HTTP server.

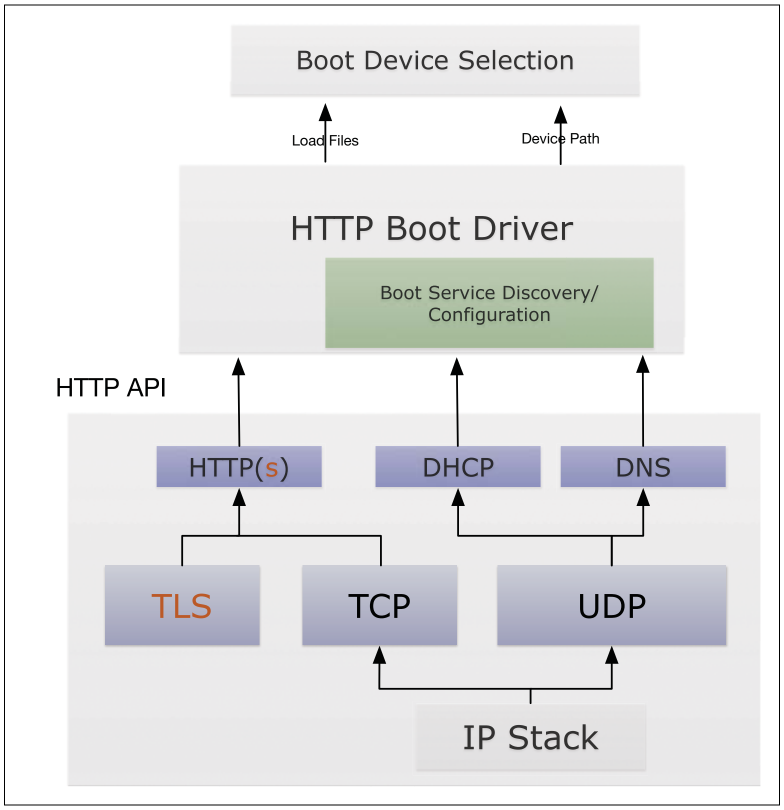


Figure 3 UEFI HTTP Boot architecture

The functionality needed in the HTTP Boot scenario is limited to client-initiated requests to download the boot files. TLS (Transport Layer Security) resources are consumed if information encryption is needed. The HTTP Boot driver produces Load Files and Device Path operations. Boot Device Selection (BDS) will provide the boot options for HTTP Boot.

Once a boot option for HTTP Boot is executed, a particular network interface is selected. HTTP Boot driver will perform all steps on that network interface.

Information exchange in HTTP Boot

In general, the HTTP Boot client acquires the network address from a DHCP server and then downloads a Network Boot Program (NBP) to execute it in the network.

The HTTP Boot information exchange sequence is:

1. The client initiates the DHCP process by broadcasting a DHCP request containing the HTTP Boot identifier. Assuming that a DHCP server with HTTP Boot extension is available, after several intermediate steps, a boot resource location in URI format will be provided to the HTTP Boot client.
2. The URI points to the NBP that is appropriate for the client's request. Then the client uses the HTTP protocol to download the NBP from the HTTP server to its memory.
3. The client executes the downloaded NBP image. This program may then use other UEFI interfaces for further system setup, based on the NBP design.

Device Path and Load Files Protocols

If both IPv4 and IPv6 are supported in the HTTP Boot environment, the HTTP Boot driver should create two child handles, with Load Files and Device Path installed on each child handle. These are described in the Unified Extensible Firmware Interface Specification (Version 2.5 Errata A).

- Device Path protocol

  An IP device path node and a boot URI device path node are appended to the parent device path message, for example:

  ```
  PciRoot(0x0)/Pci(0x19, 0x0)/MAC(01234ABCDE, 0x0)/IPv4(0.0.0.0, 0, DHCP,
  0.0.0.0, 0.0.0.0, 0.0.0.0)/Uri()

  PciRoot(0x0)/Pci(0x19, 0x0)/MAC(01234ABCDE, 0x0)/IPv6(::/128, 0, Static,
  ::/128, ::/ 128, 0)/Uri()
  ```

  The device path will be updated after retrieving the boot resource information and IP address. For example, if the NBP is a UEFI-formatted executable program, the device patch will be updated to:

  ```
  PciRoot(0x0)/Pci(0x19, 0x0)/MAC(01234ABCDE, 0x0)/IPv4(192.168.5.1, TCP, DHCP,
  192.168.5.20, 192.168.5.1,
  255.255.255.0)/Uri(http://192.168.5.1/boot/grub2/x86_64-efi/core.efi)

  PciRoot(0x0)/Pci(0x19, 0x0)/MAC(01234ABCDE, 0x0)/IPv6(2001::1, TCP,
  StatefulAutoConfigure, 2001::20, 2001::1,
  64)/Uri(http://2001::1/boot/grub2/x86_64-efi/core.efi)
  ```

  These two instances allow for the boot manager to decide a preference between IPv6 and IPv4. If the NBP is a binary image which could be mounted as a Random Access Memory (RAM) disk. The HTTP Boot driver will register the RAM disk with the downloaded NBP, by appending a RamDisk device node to the example device path above:

  ```
  PciRoot(0x0)/Pci(0x19, 0x0)/MAC(01234ABCDE, 0x0)/IPv4(192.168.5.1, TCP, DHCP,
  192.168.5.20, 192.168.5.1,
  255.255.255.0)/Uri(http://192.168.5.1/boot.iso)/RamDisk(0x049EA000, 0x5DEA000,
  0, 3D5ABD30-4175-87CE-6D64-D2ADE523C4BB)
  ```

```
PciRoot(0x0)/Pci(0x19, 0x0)/MAC(01234ABCDE, 0x0)/IPv6(2001::1, TCP,
StatefulAutoConfigure, 2001::20, 2001::1,
64)/Uri(http://2001::1/boot.iso)/RamDisk(0x049EA000, 0x5DEA000, 0,
3D5ABD30-4175-87CE-6D64-D2ADE523C4BB)
```

The boot manager could use these example device paths to match the device that produces a device path protocol, including a URI device path node in the system. The boot URI information could be retrieved from a DHCP server with the HTTP Boot extension, or from a boot option that includes a short-form URI device path, or created by a UEFI application or extracted from text entered by a user.

- Load Files Protocol

  The HTTP Boot driver expands the short-form URI device path after retrieving the IP address configuration from the DHCP server when the short-form URI device path is inputted to HTTP Boot driver through the Load Files protocol. The boot manager then enumerates all Load Files protocol instances, and invokes the Load Files protocol with the device path during the matching process.

Using HTTP Boot with ThinkSystem servers

In this section, we demonstrate how HTTP Boot can be used with Lenovo ThinkSystem servers. We also show how to establish an HTTP Boot server in SLES 12 SP2.

SLES support of HTTP Boot on ThinkSystem servers:

- SLES 12 SP2, SLES 12 SP3, SLES 12 SP4 and SLES 15 support HTTP Boot.
- ThinkSystem UEFI code released November 2018 dropped support of HTTP Boot on SLES 12 SP2.
- For SLES 12 SP3, upgrade grub2 per these guidelines:

  https://www.suse.com/support/update/announcement/2018/suse-ru-20183378-1/

In our lab, we have configured two servers: One serves as the HTTP Boot server, and the other as the HTTP Boot client. The detailed network interfaces for the HTTP Boot server are as follows:

- IPv4 address: 192.168.5.1/24
- IPv6 address: 2001:db8:f00f:cafe:1/64
- Domain name for IPv4: www.httpboot.local
- Domain name for IPv6: www.httpbootip6.local

There are two different ways to install the OS through HTTP Boot:

- The first is the ordinary one that uses grub as HTTP NBP. You have to edit the grub.cfg file, which loads the kernel and initrd from the installation image through the http protocol and kicks the installation to start.
- The second way requires firmware with Persistent Memory (PMEM) support in the UEFI that pulls the ISO file directly from the HTTP server and then stores it as a locally attached bootable CDROM device. But OS kernel support is needed to provide a native driver for that type of special PMEM device.

Both ways are implemented in SLES 12 SP2.

Configuring the HTTP Boot server

This section introduces how to setup HTTP service, DHCP service, and DNS service on one physical machine. The detailed network interface information is shown in Figure 4:

```
eth0    Link encap:Ethernet  HWaddr 40:F2:E9:68:07:D8
        inet addr:192.168.5.1  Bcast:192.168.5.255  Mask:255.255.255.0
        inet6 addr: fe80::42f2:e9ff:fe68:7d8/64 Scope:Link
        inet6 addr: 2001:db8:f00f:cafe::1/64 Scope:Global
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:456972 errors:0 dropped:0 overruns:0 frame:0
        TX packets:558334 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:28791298 (27.4 Mb)  TX bytes:486341274 (463.8 Mb)
        Memory:905a0000-905bffff
```

Figure 4   Network Interface Information

Configuring the HTTP service

The grub2 toolkit in SLES12 SP2 provides shell commands to establish an HTTP server in a simple way. You can run the following command to configure an HTTP service

`grub2-mknetdir --net-directory=DIR --modules=http`

The DIR parameter is the root directory of the HTTP server. In our example, /var/www/htdocs is used as the HTTP server root directory:

```
grub2-mknetdir --net-directory=/srv/www/htdocs --modules=http
```

Figure 5   grub2-mknetdir command

To launch the HTTP service, run the following command as root:

```
root@linux-pz68:~# systemctl start apache2
```

Figure 6   Launch the HTTP service

To make the service start automatically at boot time, use the following command:

```
root@linux-pz68:~# systemctl enable apache2
Created symlink from /etc/systemd/system/httpd.service to
/usr/lib/systemd/system/apache2.service.
Created symlink from /etc/systemd/system/apache.service to
/usr/lib/systemd/system/apache2.service.
Created symlink from /etc/systemd/system/multi-user.target.wants/apache2.service to
/usr/lib/systemd/system/apache2.service.
```

Figure 7   Start HTTP service at boot time

To check the HTTP service status, use the following command:

```
root@linux-pz68:~# systemctl status apache2 -l
apache2.service - The Apache Webserver
       Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; vendor preset: disabled)
       Active: active (running) since Mon 2017-05-22 03:02:09 EDT; 14s ago
 Main PID: 27804 (/usr/sbin/httpd)
       Status: "Total requests: 0; Current requests/sec: 0; Current traffic:   0 B/sec"
         Tasks: 6
      CGroup: /system.slice/apache2.service
                    ··27804 /usr/sbin/httpd-prefork -f /etc/apache2/httpd.conf -DSYSCONFIG -C PidFile
/var/run/httpd.pid -C Include /etc/apache2/sysconfig.d/ -DSYSTEMD -DFOREGROUND -k start
                    ··27811 /usr/sbin/httpd-prefork -f /etc/apache2/httpd.conf -DSYSCONFIG -C PidFile
/var/run/httpd.pid -C Include /etc/apache2/sysconfig.d/ -DSYSTEMD -DFOREGROUND -k start
                    ··27812 /usr/sbin/httpd-prefork -f /etc/apache2/httpd.conf -DSYSCONFIG -C PidFile
/var/run/httpd.pid -C Include /etc/apache2/sysconfig.d/ -DSYSTEMD -DFOREGROUND -k start
                    ··27813 /usr/sbin/httpd-prefork -f /etc/apache2/httpd.conf -DSYSCONFIG -C PidFile
/var/run/httpd.pid -C Include /etc/apache2/sysconfig.d/ -DSYSTEMD -DFOREGROUND -k start
                    ··27815 /usr/sbin/httpd-prefork -f /etc/apache2/httpd.conf -DSYSCONFIG -C PidFile
/var/run/httpd.pid -C Include /etc/apache2/sysconfig.d/ -DSYSTEMD -DFOREGROUND -k start
                    ··27816 /usr/sbin/httpd-prefork -f /etc/apache2/httpd.conf -DSYSCONFIG -C PidFile
/var/run/httpd.pid -C Include /etc/apache2/sysconfig.d/ -DSYSTEMD -DFOREGROUND -k start

May 22 03:02:09 linux-pz68 systemd[1]: Starting The Apache Webserver...
May 22 03:02:09 linux-pz68 systemd[1]: Started The Apache Webserver.
```

Figure 8 Checking the status of the HTTP service

If the status of your HTTP service is dead or failure instead of running, follow the warning message in the terminal to check the reason for the failure.

Configuring the DNS service

The DNS service is optional for the HTTP Boot server. If you want to use the easy-to-remember domain names instead of numeric IP addresses in the HTTP server, a DNS service should be set up.

Use the following steps to configure the DNS service to your reference:

1. Add the domain zones by editing the named.conf file under the /etc directory. Add the following forward lookup zone entries to the existing file.

```
zone "httpboot.local" in {
   allow-transfer { any; };
   file "master/httpboot.local";
   type master;
};

zone "httpbootip6.local" in {
   allow-transfer { any; };
   file "master/httpbootip6.local";
   type master;
};
```

Figure 9 Additions to the named.conf file

2. Navigate to the /var/lib/named/master directory and create two blank files with the following names:

   – httpboot.local
   – httpbootip6.local

3. Edit the content of the files.
   - Add the following information in the httpboot.local file:

```
$TTL 2d
@      IN SOA linux-pz68.labs.lenovo.com.root.linux-pz68.labs.lenovo.com. (
                2017032400; serial
                3h        ; refresh
                1h        ; retry
                1w        ; expiry
                1d )      ; minimum

httpboot.local.  IN NS  www.httpboot.local.
httpboot.local.  IN A   192.168.5.1
www              IN A   192.168.5.1
```

Figure 10 Contents of the httpboot.local file

   - Add the following information in the httpbootip6.local file:

```
$TTL 2d
@      IN SOA    linux-pz68.labs.lenovo.com.     root.linux-pz68.labs.lenovo.com. (
                      2017052301      ; serial
                      3h              ; refresh
                      1h              ; retry
                      1w              ; expiry
                      1d )            ; minimum

httpbootip6.local.    IN NS     www.httpbootip6.local.
httpbootip6.local.    IN AAAA   2001:db8:f00f:cafe::1
www                   IN AAAA   2001:db8:f00f:cafe::1
```

Figure 11 Contents of the httpbootip6.local file

4. Launch the DNS service using the following command:

```
root@linux-pz68:~# systemctl start named
```

Figure 12 Launch the DNS service

5. Check the DNS service status using the following command:

```
root@linux-pz68:~# systemctl status named -l
● named.service – LSB: Domain Name System (DNS) server, named
        Loaded: loaded (/etc/init.d/named; bad; vendor preset: disabled)
        Active: active (running) since Mon 2017-05-22 03:05:09 EDT; 14s ago
             Docs: man: systemd-sysv-generator(8)
            Tasks: 15 (limit: 512)
        CGroup: /system.slice/named.service
                      ··17250 /usr/sbin/named -t /var/lib/named -u named
```

Figure 13 Check the status of the DNS service

If the status of your DNS service is dead or fail instead of running, follow the warning message in the terminal to check the reason causing the failure. Ping the domain name to check if the DNS service can run successfully:

```
root@linux-pz68:~# ping httpboot.local
PING httpboot.local (192.168.5.1) 56(84) bytes of data.
64 bytes from 192.168.5.1: icmp_seq=1 ttl=64 time=0.026 ms
64 bytes from 192.168.5.1: icmp_seq=2 ttl=64 time=0.038 ms
64 bytes from 192.168.5.1: icmp_seq=3 ttl=64 time=0.020 ms

root@linux-pz68:~# ping6 httpbootip6.local
PING httpbootip6.local(2001:db8:f00f:cafe::1) 56 data bytes
64 bytes from 2001:db8:f00f:cafe::1: icmp_seq=1 ttl=64 time=0.019 ms
64 bytes from 2001:db8:f00f:cafe::1: icmp_seq=2 ttl=64 time=0.019 ms
64 bytes from 2001:db8:f00f:cafe::1: icmp_seq=3 ttl=64 time=0.020 ms
```

Figure 14 Ping commands to check the DNS service

Configuring DHCP Service

The dhcp package contains an Internet Systems Consortium (ISC) DHCP server. Install the package as root, and then modify the dhcpd service configuration files – dhcpd.conf for IPv4 and dhcpd6.conf for IPv6 under the /etc directory:

DHCP service for IPv4

1. Edit the dhcpd.conf file as follows:

```
option domain-name "httpboot.local";
option domain-name-servers 192.168.5.1;
default-lease-time 14400;
subnet 192.168.5.0 netmask 255.255.255.0 {
      range dynamic-bootp 192.168.5.20 192.168.5.99;
      default-lease-time 14400;
      max-lease-time 172800;
      class "pxeclients" {
            match if substring (option vendor-class-identifier, 0, 9) = "PXEClient";
            filename "Bootx64.efi";
       }
      class "httpclients" {
            option vendor-class-identifier "HTTPClient";
            match if substring (option vendor-class-identifier, 0, 10) =
"HTTPClient";
            filename "http://www.httpboot.local/boot/grub2/x86_64-efi/core.efi";
      }
}
```

Figure 15 Additions to the dhcpd.conf file

2. Start the DHCP service, using the following command:

```
root@linux-pz68:~#  systemctl start dhcpd.service
```

Figure 16 Start the DHCP service

3. Make sure the service start automatically at boot time by using the following command:

```
root@linux-pz68:~# systemctl enable dhcpd
Created symlink from /etc/systemd/system/dhcp-server.service to
/usr/lib/systemd/system/dhcpd.service.
Created symlink from /etc/systemd/system/multi-user.target.wants/dhcpd.service to
/usr/lib/systemd/system/dhcpd.service.
```

Figure 17 Start the DHCP service at boot time

4. Check the status of the DHCP service for IPv4 using the following command:

```
root@linux-pz68:~# systemctl status dhcpd -l
● dhcpd.service - ISC DHCPv4 Server
   Loaded: loaded (/usr/lib/systemd/system/dhcpd.service; enabled; vendor preset:
disabled)
   Active: active (running) since Mon 2017-05-22 03:02:34 EDT; 7s ago
  Process: 27823 ExecStart=/usr/lib/dhcp/dhcpd -4 start (code=exited,
status=0/SUCCESS)
 Main PID: 27959 (dhcpd)
    Tasks: 1 (limit: 512)
   CGroup: /system.slice/dhcpd.service
           ··27959 /usr/sbin/dhcpd -4 -cf /etc/dhcpd.conf -pf /var/run/dhcpd.pid
-chroot /var/lib/dhcp -lf /db/dhcpd.leases -user dhcpd -group nogroup eth0

May 22 03:02:34 linux-pz68 dhcpd[27958]: Not searching LDAP since ldap-server,
ldap-port and ldap-base-dn were not specified in the config file
May 22 03:02:34 linux-pz68 dhcpd[27958]: Config file: /etc/dhcpd.conf
May 22 03:02:34 linux-pz68 dhcpd[27958]: Database file: /db/dhcpd.leases
May 22 03:02:34 linux-pz68 dhcpd[27958]: PID file: /var/run/dhcpd.pid
May 22 03:02:34 linux-pz68 dhcpd[27958]: Wrote 0 class decls to leases file.
May 22 03:02:34 linux-pz68 dhcpd[27958]: Wrote 1 leases to leases file.
May 22 03:02:34 linux-pz68 dhcpd[27958]: Listening on
LPF/eth0/40:f2:e9:68:07:d8/192.168.5.0/24
May 22 03:02:34 linux-pz68 dhcpd[27959]: Server starting service.
May 22 03:02:34 linux-pz68 dhcpd[27823]: Starting ISC DHCPv4 Server [chroot]..done
May 22 03:02:34 linux-pz68 systemd[1]: Started ISC DHCPv4 Server.
```

Figure 18 Check the status of the DHCP service

If the status of your DHCP service for IPv4 is dead or fail instead of running, follow the warning message in the terminal to check the reason causing the failure.

DHCP service for IPv6

1. Edit the dhcpd6.conf file as follows:

```
option dhcp6.bootfile-url code 59 = string;
option dhcp6.vendor-class code 16 = {integer 32, integer 16, string};

subnet6 2001:db8:f00f:cafe::/64 {
   range6 2001:db8:f00f:cafe::42:10 2001:db8:f00f:cafe::42:99;
   option dhcp6.domain-search "httpbootip6.local";
   option dhcp6.name-servers 2001:db8:f00f:cafe::1;

   class "PXEClient" {
      match substring (option dhcp6.vendor-class, 6, 9);
              }
              subclass "PXEClient" "PXEClient" {
                       option dhcp6.bootfile-url
"tftp://[2001:db8:f00f:cafe::1]/Bootx64.efi";
              }

               class "HTTPClient" {
                             match substring (option dhcp6.vendor-class, 6, 10);
              }
              subclass "HTTPClient" "HTTPClient" {
                      option dhcp6.bootfile-url
"http://www.httpbootip6.local/boot/grub2/x86_64-efi/core.efi";
                      option dhcp6.vendor-class 0 10 "HTTPClient";
              }
}
```

Figure 19 Additions to the dhcpd6.conf file

2. To start the DHCP service, use the following command:

```
root@linux-pz68:~#  systemctl start dhcpd6.service
```

Figure 20 Start the DHCP IPv6 service

3. To make the service start automatically at boot time, use the following command:

```
root@linux-pz68:~# systemctl enable dhcpd6
Created symlink from /etc/systemd/system/dhcp6-server.service to
/usr/lib/systemd/system/dhcpd6.service.
Created symlink from /etc/systemd/system/multi-user.target.wants/dhcpd6.service to
/usr/lib/systemd/system/dhcpd6.service.
```

Figure 21 Configure the DHCP IPv6 service to start at boot time

4. Check the status of the DHCP service for IPv6 using the following command:

```
root@linux-pz68:~# systemctl status dhcpd6 -l
• dhcpd6.service - ISC DHCPv6 Server
   Loaded: loaded (/usr/lib/systemd/system/dhcpd6.service; enabled; vendor preset:
disabled)
   Active: active (running) since Mon 2017-05-22 03:02:50 EDT; 3s ago
  Process: 27974 ExecStart=/usr/lib/dhcp/dhcpd -6 start (code=exited,
status=0/SUCCESS)
 Main PID: 28109 (dhcpd)
    Tasks: 1 (limit: 512)
   CGroup: /system.slice/dhcpd6.service
           ··28109 /usr/sbin/dhcpd -6 -cf /etc/dhcpd6.conf -pf /var/run/dhcpd6.pid
-chroot /var/lib/dhcp6 -lf /db/dhcpd6.leases -user dhcpd -group nogroup eth0

May 22 03:02:50 linux-pz68 dhcpd[28108]: Config file: /etc/dhcpd6.conf
May 22 03:02:50 linux-pz68 dhcpd[28108]: Database file: /db/dhcpd6.leases
May 22 03:02:50 linux-pz68 dhcpd[28108]: PID file: /var/run/dhcpd6.pid
May 22 03:02:50 linux-pz68 dhcpd[28108]: Wrote 0 class decls to leases file.
May 22 03:02:50 linux-pz68 dhcpd[28108]: Wrote 0 NA, 0 TA, 0 PD leases to lease file.
May 22 03:02:50 linux-pz68 dhcpd[28108]: Bound to *:547
May 22 03:02:50 linux-pz68 dhcpd[28108]: Listening on
Socket/5/eth0/2001:db8:f00f:cafe::/64
May 22 03:02:50 linux-pz68 dhcpd[28109]: Server starting service.
May 22 03:02:50 linux-pz68 dhcpd[27974]: Starting ISC DHCPv6 Server [chroot]..done
May 22 03:02:50 linux-pz68 systemd[1]: Started ISC DHCPv6 Server.
```

Figure 22 Check the DHCP IPv6 service

If the status of your DHCP service for IPv6 is dead or fail instead of running, you could follow the warning message in the terminal to check the reason causing the failure.

Configuring HTTP Boot Server File Catalog

1. Create a folder called ISO under the /srv/www directory for placing OS images.
2. Create a folder called sles12sp2 under the /srv/www/htdocs directory as the OS image mounting point.
3. Modify the grub.cfg file under the /srv/www/htdocs/boot/grub2 directory as follows.

```
set timeout=8
menuentry "Install SLE12 SP2 (UEFI)"{
     linuxefi /sles12sp2/boot/x86_64/loader/linux
install=http://www.httpboot.local/sles12sp2
     initrdefi /sles12sp2/boot/x86_64/loader/initrd
}
```

Figure 23 Changes to the grub.cfg file

4. Mount the SLES12 SP2 image to the /sles12sp2 mounting point.

The entire HTTP Boot server directory topology is illustrated as below:

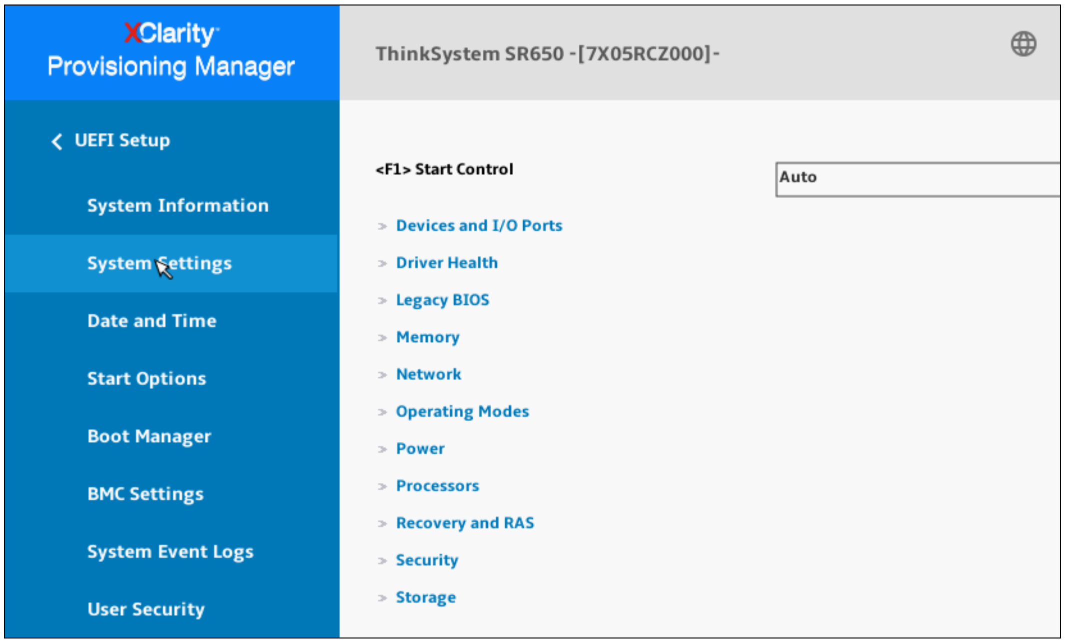
```
root@linux-pz68:~# tree -L 2 /srv/www
/srv/www
··· ISO
·   ··· RHEL-7.3.iso
·   ··· SLES-12-SP2.iso
··· cgi-bin
··· htdocs
·   ··· boot
·   ··· favicon.ico
·   ··· index.html
·   ··· repo
·   ··· rhel73
·   ··· robots.txt
·   ··· sles12sp2
··· perl-lib
    ··· NU
    ··· SMT
```

Figure 24 HTTP Boot server directory tree

Configuring HTTP Boot Client

HTTP Boot should be supported and enabled on the UEFI HTTP Boot clients in the BIOS.

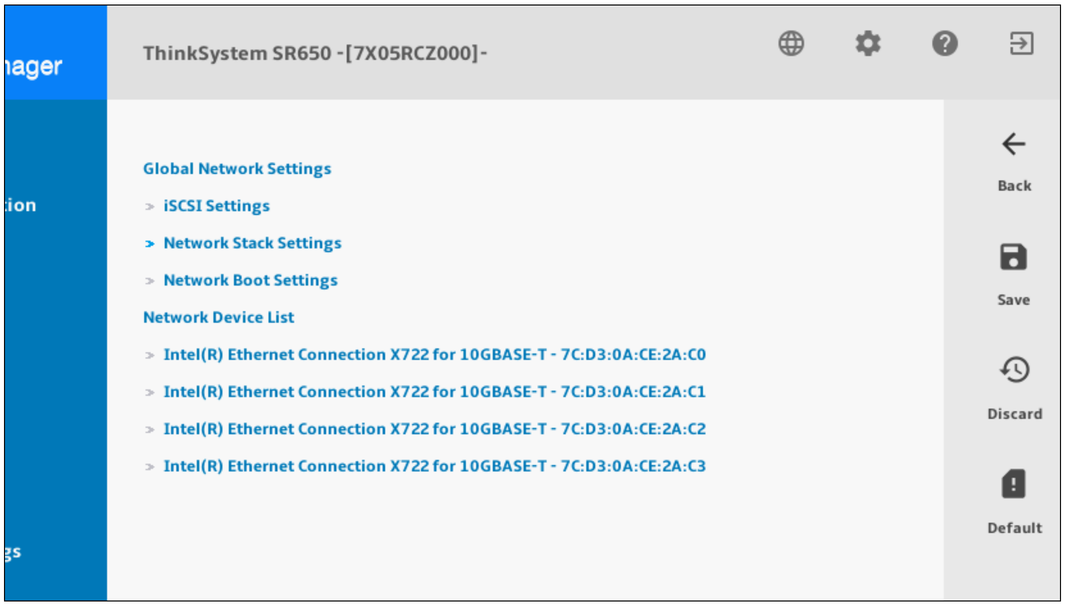
1. Power on the server and press the F1 key to enter the UEFI Setup menu.



<F1> Start Control Auto

Figure 25 UEFI System Setup menu for ThinkSystem servers

2. Select **System Settings** → **Network**, Figure 26 on page 16.

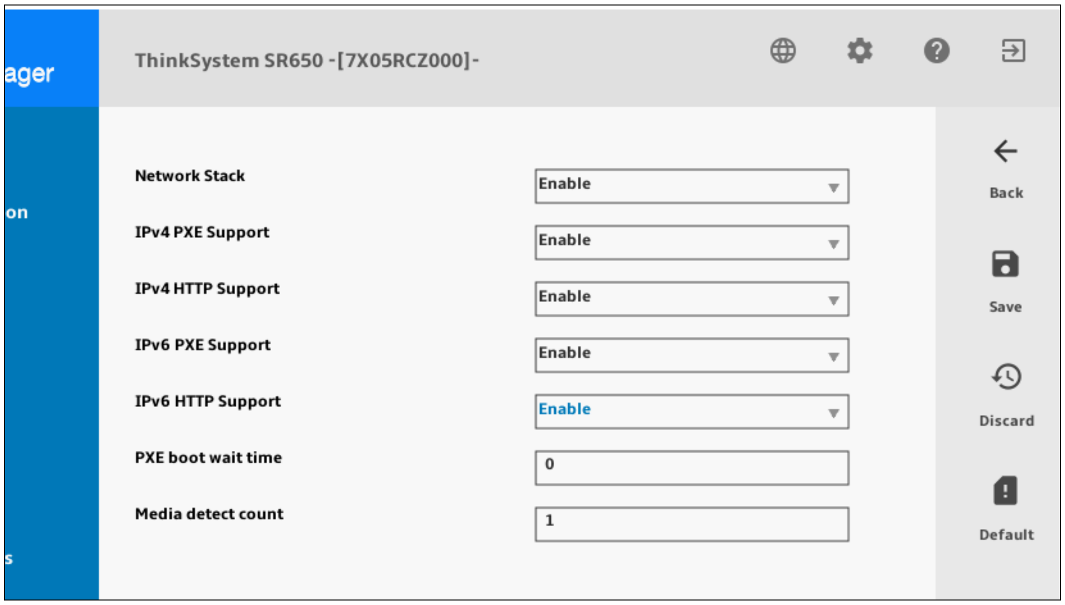


Global Network Settings
- iSCSI Settings
- Network Stack Settings
- Network Boot Settings

Network Device List
- Intel(R) Ethernet Connection X722 for 10GBASE-T - 7C:D3:0A:CE:2A:C0
- Intel(R) Ethernet Connection X722 for 10GBASE-T - 7C:D3:0A:CE:2A:C1
- Intel(R) Ethernet Connection X722 for 10GBASE-T - 7C:D3:0A:CE:2A:C2
- Intel(R) Ethernet Connection X722 for 10GBASE-T - 7C:D3:0A:CE:2A:C3

Figure 26 Network Setup Menu

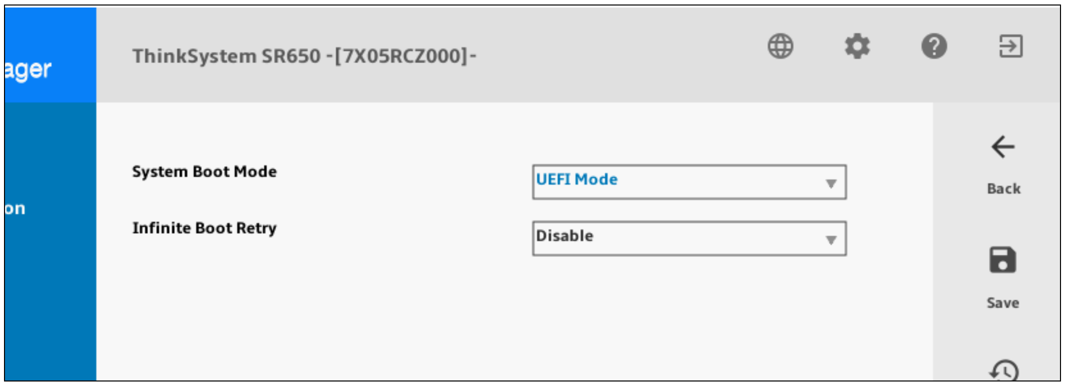
3. Select Network Stack Settings. Then enable IPv4 HTTP Support and IPv6 HTTP Support, Figure 27.



Network Stack	Enable
IPv4 PXE Support	Enable
IPv4 HTTP Support	Enable
IPv6 PXE Support	Enable
IPv6 HTTP Support	Enable
PXE boot wait time	0
Media detect count	1

Figure 27 Network Stack Settings

4. Select **Boot Manager** → **Boot Modes**. Set System Boot Mode to UEFI Mode, per Figure 28 on page 17.



System Boot Mode: UEFI Mode
Infinite Boot Retry: Disable

Figure 28   Boot Mode Settings

5. Save the configuration and exit the UEFI Setup menu.

Installing an OS through HTTP Boot

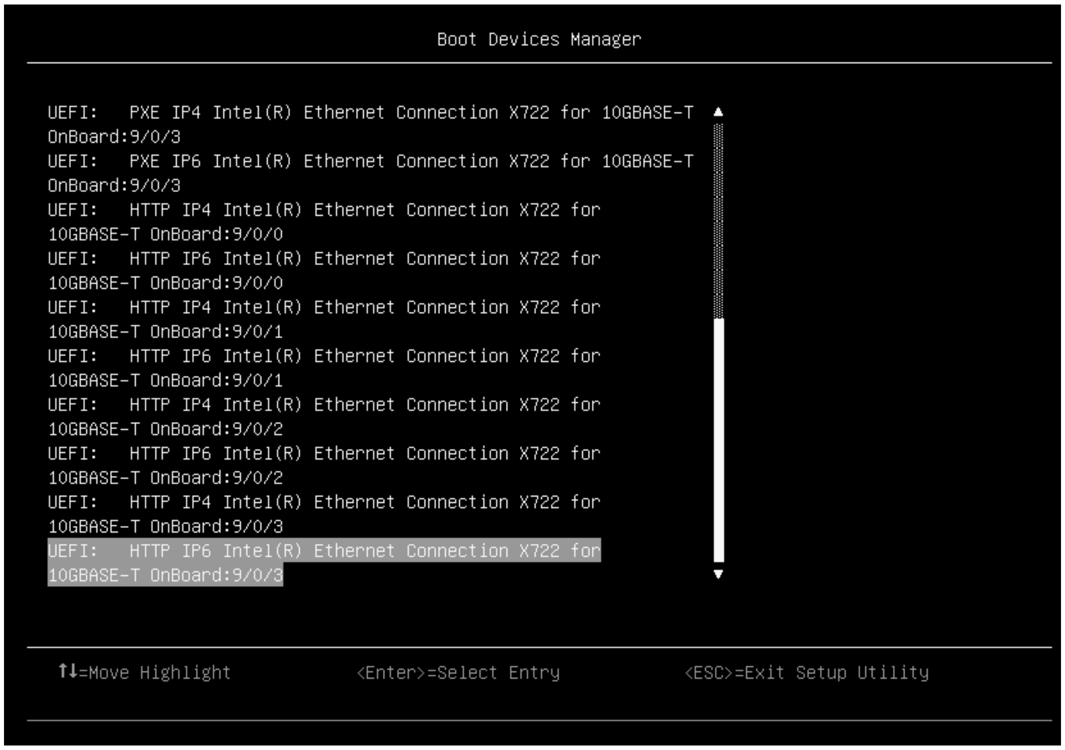
Tip: You can perform these tasks at the local console in front of the server, or if you have XClarity Controller Advanced on your ThinkSystem server, you can perform them remotely using the remove KVM functionality of XClarity Controller.

Before you install the OS through HTTP Boot, verify the following:

- The HTTP Boot server and client machine are in the same network segment.
- The HTTP service and DHCP service are in active status.
- The HTTP Boot option is enabled and the boot mode is set to UEFI in the BIOS on the client side.

To install OS through HTTP Boot on the server, perform the following steps:

1. Power on the Lenovo ThinkSystem server and press F12 to enter the One Time Boot Device menu, Figure 29 on page 18.

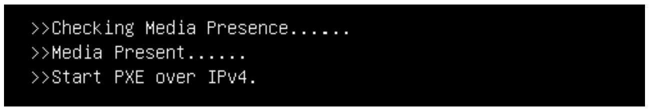


```
Boot Devices Manager

UEFI:   PXE IP4 Intel(R) Ethernet Connection X722 for 10GBASE-T
OnBoard:9/0/3
UEFI:   PXE IP6 Intel(R) Ethernet Connection X722 for 10GBASE-T
OnBoard:9/0/3
UEFI:   HTTP IP4 Intel(R) Ethernet Connection X722 for
10GBASE-T OnBoard:9/0/0
UEFI:   HTTP IP6 Intel(R) Ethernet Connection X722 for
10GBASE-T OnBoard:9/0/0
UEFI:   HTTP IP4 Intel(R) Ethernet Connection X722 for
10GBASE-T OnBoard:9/0/1
UEFI:   HTTP IP6 Intel(R) Ethernet Connection X722 for
10GBASE-T OnBoard:9/0/1
UEFI:   HTTP IP4 Intel(R) Ethernet Connection X722 for
10GBASE-T OnBoard:9/0/2
UEFI:   HTTP IP6 Intel(R) Ethernet Connection X722 for
10GBASE-T OnBoard:9/0/2
UEFI:   HTTP IP4 Intel(R) Ethernet Connection X722 for
10GBASE-T OnBoard:9/0/3
UEFI:   HTTP IP6 Intel(R) Ethernet Connection X722 for
10GBASE-T OnBoard:9/0/3

↑↓=Move Highlight        <Enter>=Select Entry        <ESC>=Exit Setup Utility
```

Figure 29   One Time Boot Device List

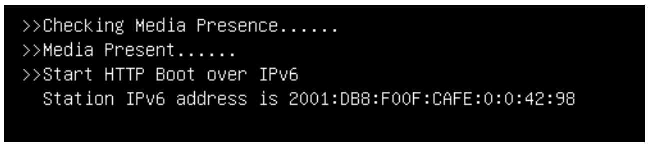
2. Select the network interface connected to the HTTP Boot server and press Enter to launch HTTP Boot. The HTTP Boot would download the NBP over IPv4 or IPv6 protocol based on your choice.

   The boot processes in IPv4 protocol and IPv6 protocol are shown in Figure 30 and Figure 31 respectively. The NBP will be executed subsequently, as shown in Figure 32.



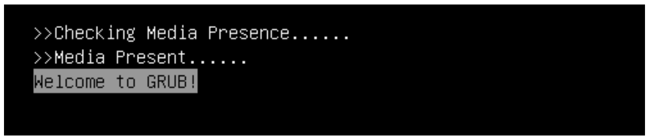
```
>>Checking Media Presence......
>>Media Present......
>>Start PXE over IPv4.
```

Figure 30   HTTP Boot over IPv4 Initialization



```
>>Checking Media Presence......
>>Media Present......
>>Start HTTP Boot over IPv6
  Station IPv6 address is 2001:DB8:F00F:CAFE:0:0:42:98
```

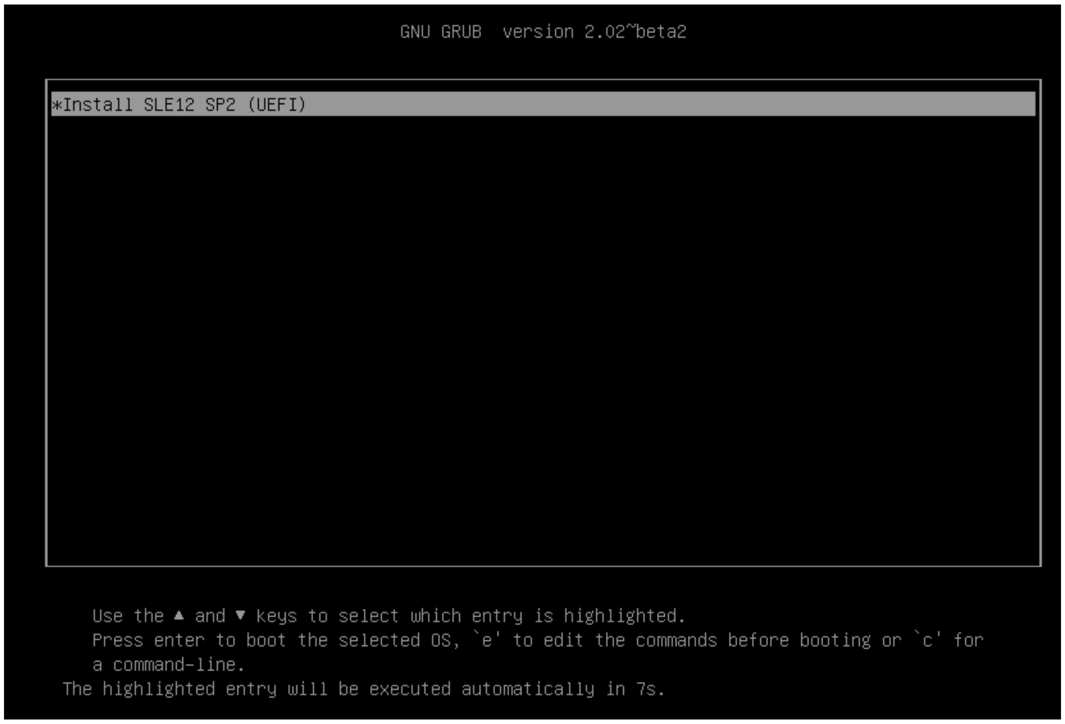
Figure 31   HTTP Boot over IPv6 initialization



```
>>Checking Media Presence......
>>Media Present......
Welcome to GRUB!
```

Figure 32   Download and Run the NBP

3. The HTTP Boot menu of OS installation modified in “Configuring HTTP Boot Server File Catalog” on page 14 appears as shown in Figure 33 on page 19 on the HTTP Boot client.



```
                    GNU GRUB  version 2.02~beta2

 *Install SLE12 SP2 (UEFI)

     Use the ▲ and ▼ keys to select which entry is highlighted.
     Press enter to boot the selected OS, `e' to edit the commands before booting or `c' for
     a command-line.
  The highlighted entry will be executed automatically in 7s.
```

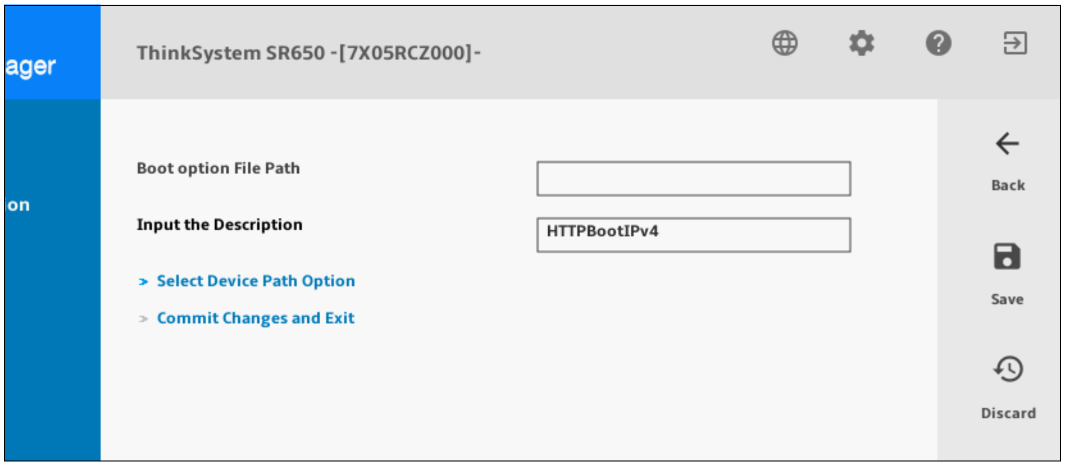
Figure 33   HTTP Boot Menu of OS Installation

4. Choose the OS to be installed on your HTTP Boot client.

Creating a Boot Option

You can create the boot option of HTTP Boot in the UEFI Setup menu if you prefer an easy-to-remember boot option. You can create both IPv4 and IPv6 HTTP Boot options. In our example, we create the IPv4 HTTP Boot option:

1. Power on the server and press F1 to enter the UEFI Setup menu.
2. Select **Boot Manager** → **Add UEFI Full Path Boot Option**. Input the boot option description. In this document, "HTTPBootIPv4" is used as an example, Figure 34.



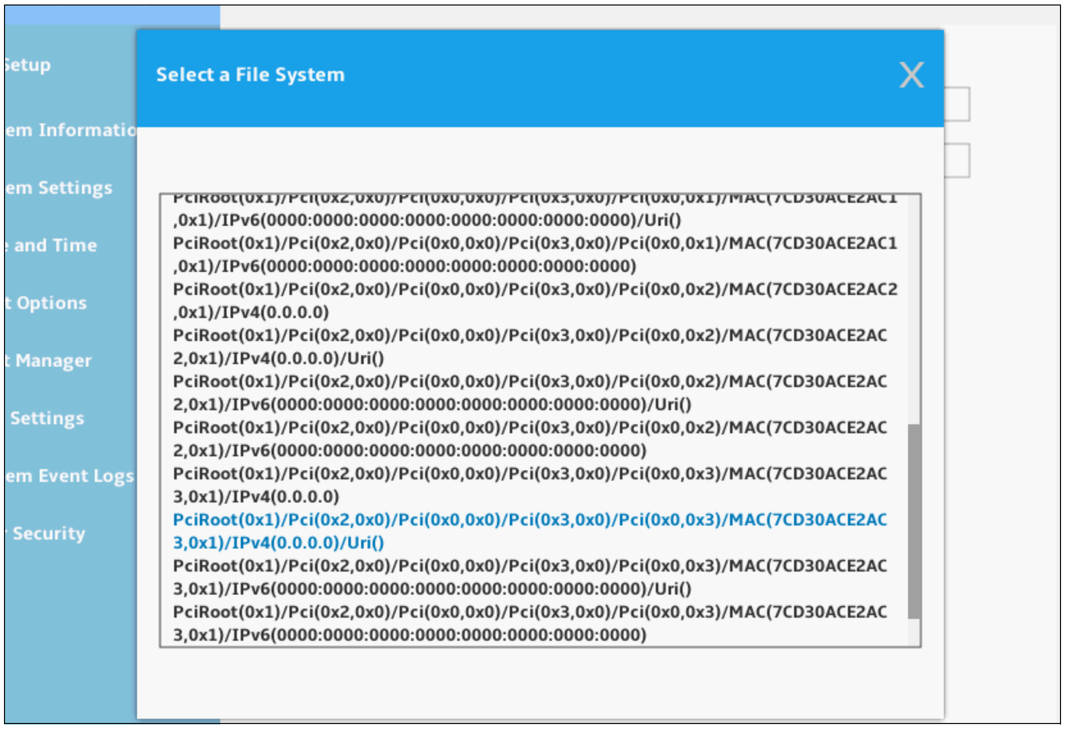
ThinkSystem SR650 -[7X05RCZ000]-

Boot option File Path

Input the Description: HTTPBootIPv4

Figure 34   Input Boot Option Description

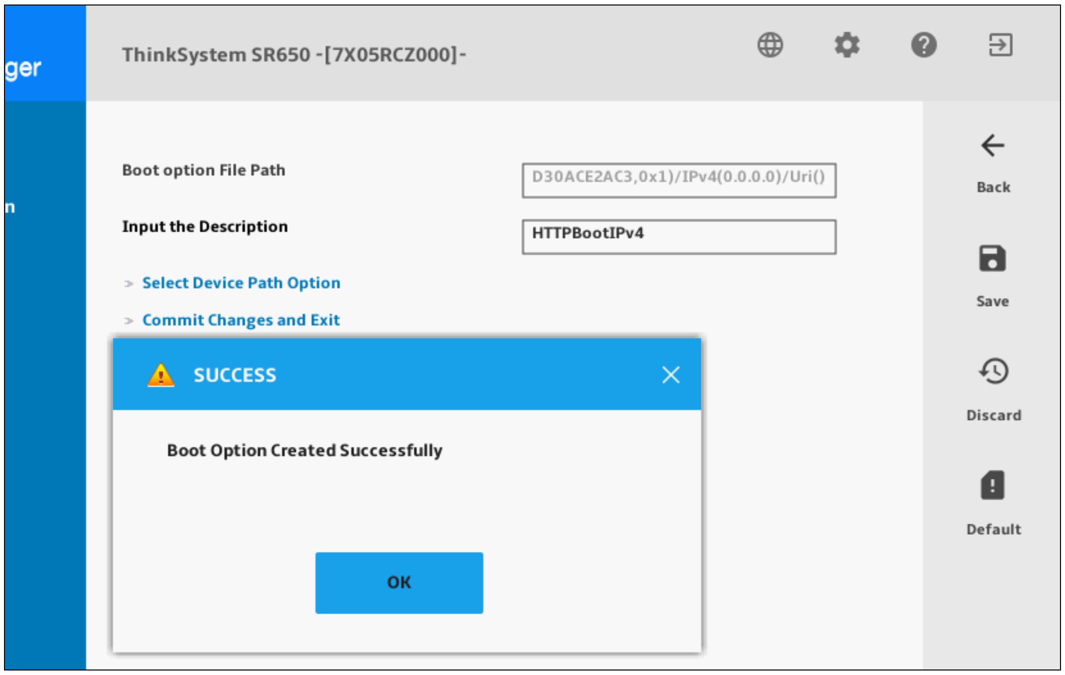
3. Select the HTTP Boot network interface in Select Device Path Option, Figure 35.



```
PciRoot(0x1)/Pci(0x2,0x0)/Pci(0x0,0x0)/Pci(0x3,0x0)/Pci(0x0,0x1)/MAC(7CD30ACE2AC1
,0x1)/IPv6(0000:0000:0000:0000:0000:0000:0000:0000)/Uri()
PciRoot(0x1)/Pci(0x2,0x0)/Pci(0x0,0x0)/Pci(0x3,0x0)/Pci(0x0,0x1)/MAC(7CD30ACE2AC1
,0x1)/IPv6(0000:0000:0000:0000:0000:0000:0000:0000)
PciRoot(0x1)/Pci(0x2,0x0)/Pci(0x0,0x0)/Pci(0x3,0x0)/Pci(0x0,0x2)/MAC(7CD30ACE2AC2
,0x1)/IPv4(0.0.0.0)
PciRoot(0x1)/Pci(0x2,0x0)/Pci(0x0,0x0)/Pci(0x3,0x0)/Pci(0x0,0x2)/MAC(7CD30ACE2AC
2,0x1)/IPv4(0.0.0.0)/Uri()
PciRoot(0x1)/Pci(0x2,0x0)/Pci(0x0,0x0)/Pci(0x3,0x0)/Pci(0x0,0x2)/MAC(7CD30ACE2AC
2,0x1)/IPv6(0000:0000:0000:0000:0000:0000:0000:0000)/Uri()
PciRoot(0x1)/Pci(0x2,0x0)/Pci(0x0,0x0)/Pci(0x3,0x0)/Pci(0x0,0x2)/MAC(7CD30ACE2AC
2,0x1)/IPv6(0000:0000:0000:0000:0000:0000:0000:0000)
PciRoot(0x1)/Pci(0x2,0x0)/Pci(0x0,0x0)/Pci(0x3,0x0)/Pci(0x0,0x3)/MAC(7CD30ACE2AC
3,0x1)/IPv4(0.0.0.0)
PciRoot(0x1)/Pci(0x2,0x0)/Pci(0x0,0x0)/Pci(0x3,0x0)/Pci(0x0,0x3)/MAC(7CD30ACE2AC
3,0x1)/IPv4(0.0.0.0)/Uri()
PciRoot(0x1)/Pci(0x2,0x0)/Pci(0x0,0x0)/Pci(0x3,0x0)/Pci(0x0,0x3)/MAC(7CD30ACE2AC
3,0x1)/IPv6(0000:0000:0000:0000:0000:0000:0000:0000)/Uri()
PciRoot(0x1)/Pci(0x2,0x0)/Pci(0x0,0x0)/Pci(0x3,0x0)/Pci(0x0,0x3)/MAC(7CD30ACE2AC
3,0x1)/IPv6(0000:0000:0000:0000:0000:0000:0000:0000)
```

Figure 35   Add the Boot Path

4. Select **Commit Changes and Exit** to complete the operation, Figure 36.



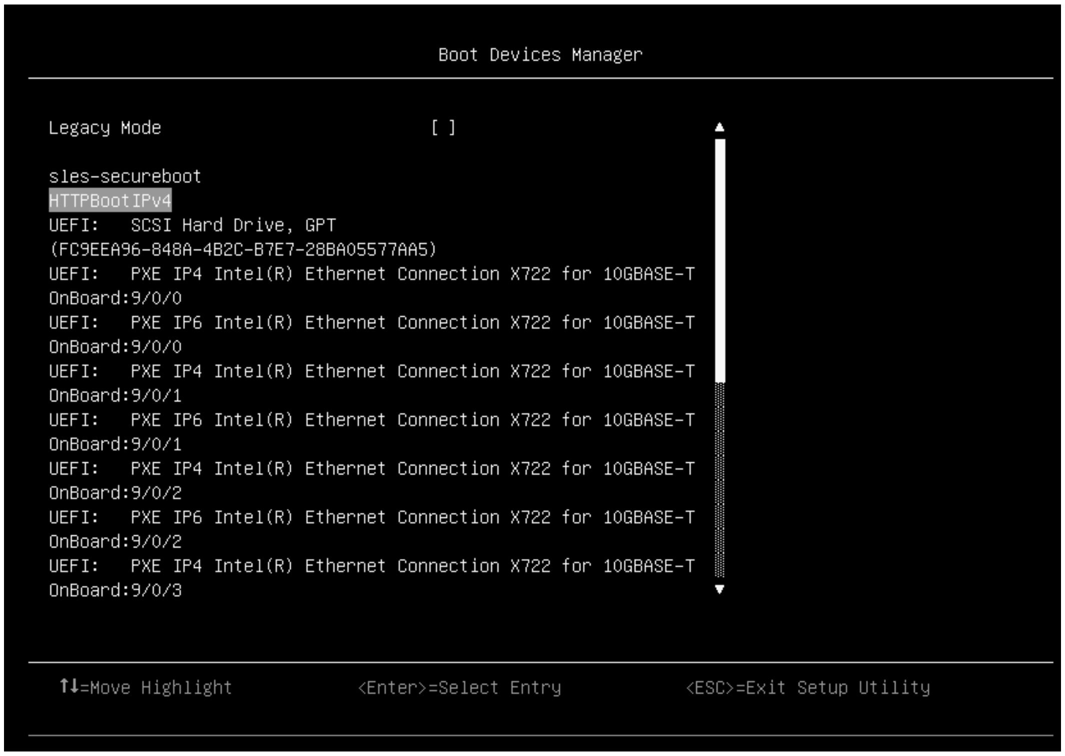
Boot option File Path: D30ACE2AC3,0x1)/IPv4(0.0.0.0)/Uri()

Input the Description: HTTPBootIPv4

Boot Option Created Successfully

Figure 36   Commit changes

5. Save the UEFI configuration and exit the UEFI Setup menu. Then press F12 to enter the One Time Boot Device menu after server reboot. You can find HTTPBootIPv4 in the boot option menu, Figure 37 on page 21.



```
                                Boot Devices Manager

  Legacy Mode                        [ ]

  sles-secureboot
  HTTPBootIPv4
  UEFI:   SCSI Hard Drive, GPT
  (FC9EEA96-848A-4B2C-B7E7-28BA05577AA5)
  UEFI:   PXE IP4 Intel(R) Ethernet Connection X722 for 10GBASE-T
  OnBoard:9/0/0
  UEFI:   PXE IP6 Intel(R) Ethernet Connection X722 for 10GBASE-T
  OnBoard:9/0/0
  UEFI:   PXE IP4 Intel(R) Ethernet Connection X722 for 10GBASE-T
  OnBoard:9/0/1
  UEFI:   PXE IP6 Intel(R) Ethernet Connection X722 for 10GBASE-T
  OnBoard:9/0/1
  UEFI:   PXE IP4 Intel(R) Ethernet Connection X722 for 10GBASE-T
  OnBoard:9/0/2
  UEFI:   PXE IP6 Intel(R) Ethernet Connection X722 for 10GBASE-T
  OnBoard:9/0/2
  UEFI:   PXE IP4 Intel(R) Ethernet Connection X722 for 10GBASE-T
  OnBoard:9/0/3

   ↑↓=Move Highlight          <Enter>=Select Entry          <ESC>=Exit Setup Utility
```

Figure 37 One Time Boot Device List with custom boot option

6. Select HTTPBootIPv4 to start the HTTP Boot process.

Acronyms

BDS	Boot Device Selection
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
HTTP	Hypertext Transfer Protocol
NBP	Network Boot Program
NIC	Network interface card
OS	Operating system
PMEM	Persistent Memory
PXE	Preboot Execution Environment
RAM	Random-access memory
TFTP	Trivial File Transfer Protocol
TLS	Transport Layer Security
URI	Uniform Resource Identifier

Change history

- May 2019:
  - Added a note regarding SLES support of HTTP Boot with SLES 12.x and SLES 15 and the dropping of HTTP Boot support with SLES 12.2 with UEFI firmware released November 2019. See "Using HTTP Boot with ThinkSystem servers" on page 7.
- November 17, 2017:
  - Grammar and readability corrections

Author

Neo Cui is a Linux Engineer of the Lenovo Data Center Group in Beijing, China. He joined the OS team in Lenovo after graduating from Ocean University of China with a research field of Ultra Wide-Band Communications. His major focus is the security and RAS feature of Linux kernel development in Lenovo.

Thanks to the following people for their contributions to this project:

- David Watts, Lenovo Press
- Jerry Tang, Information Development
- Mark Cowley, SUSE Project Manager
- Samer El Haj Mahmoud, Lenovo Lead Architect
- Mark T. Chapman, Lenovo Editor

Notices

Lenovo may not offer the products, services, or features discussed in this document in all countries. Consult your local Lenovo representative for information on the products and services currently available in your area. Any reference to a Lenovo product, program, or service is not intended to state or imply that only that Lenovo product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any Lenovo intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any other product, program, or service.

Lenovo may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

Lenovo (United States), Inc.
1009 Think Place - Building One
Morrisville, NC 27560
U.S.A.
Attention: Lenovo Director of Licensing

LENOVO PROVIDES THIS PUBLICATION “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. Lenovo may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

The products described in this document are not intended for use in implantation or other life support applications where malfunction may result in injury or death to persons. The information contained in this document does not affect or change Lenovo product specifications or warranties. Nothing in this document shall operate as an express or implied license or indemnity under the intellectual property rights of Lenovo or third parties. All information contained in this document was obtained in specific environments and is presented as an illustration. The result obtained in other operating environments may vary.

Lenovo may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Any references in this publication to non-Lenovo Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this Lenovo product, and use of those Web sites is at your own risk.

Any performance data contained herein was determined in a controlled environment. Therefore, the result obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

This document was created or updated on May 14, 2019.

Send us your comments via the **Rate & Provide Feedback** form found at
http://lenovopress.com/lp0736

Trademarks