PXE Boot over the Internet

Use cases

- Boot rescue images anywhere
- Centralise systems

Pros

- No disks onsite
- No data onsite
- Flexibility
- Lower costs on maintenance/hardware/etc.

Cons

- Need the bandwidth & low latency
- Slightly slower

How

1. Custom iPXE image w/script
2. pfSense loads script to PXE boot
3. LTSP image provides operating system
4. thinlinc fast remote workstations (optional)

Alternatives

- Onsite PXE boot (maintain another machine)
- Thinstation - instead of LTSP (hard to config/build/maintain)

Hardware

Thinclients	Shuttle DS77U7 i7 + 16GB RAM + Dell 24in screen x 2
Firewall	pfSense on Netgate SG-8860 (8 core, 2xSSD RAID 1)
Network	Enterprise Ethernet NBN 250/250mbps
VPS	Vultr Cloud Compute 8GB RAM, 160GB SSD, 4TB transfer

1. Custom iPXE image

Create VM

- Virt Manager
- Debian 11 64bit
- 2GB RAM, 2CPU, 10GB HD

Install Docker

```
wget http://get.docker.io/ -O - | sh
```

Add [user] to docker group and re-login.

Install iPXE-buildweb

Install from git repo

https://github.com/xbgmsharp/ipxe-buildweb

```
docker build --rm=true --no-cache=true -t xbgmsharp/ipxe-buildweb github.com/xbgmsharp/ipxe-buildweb.git
```

Run

```
docker run -i --rm -p 23:22 -p 8080:80 -t xbgmsharp/ipxe-buildweb /bin/bash
```

Connect

Connect to web interface within VM using Docker IP EG: http://172.17.0.2

NOTE: use 'ip a' to find docker ip

Fill out form

1. Select Standard
2. Choose output format as "UNDI only (.kpxe)"

Enter script

1. Enter script

   ```
   #!ipxe
   dhcp
   chain http://ltsp.example.com.au/ltsp/ltsp.ipxe
   ```

2. Click proceed & save undionly.kpxe

NOTE: ltsp.example.com.au is your VM where we are hosting another linux box with LTSP images. This could also be an IP address. Cannot use https!

2. pfSense setup

This assumes you have a pfSense box but other firewalls at this level will have these abilities.

NOTE: You cannot use a 'consumer level' router for this.

Setup tftp

Install tftp addon through System -> Packages

Transfer undionly build

On build VM in the downloads directory

```
scp undionly.kpxe root@<firewall IP>:/tftpboot
```

Config DHCP

Configure DHCP -> Network Boot settings;

Next Server	192.168.200.1
Default BIOS file name	undionly.kpxe

Click 'save' & 'apply'

3. LTSP build (basic)

Install LTSP

```
sudo apt install --install-recommends ltsp dnsmasq nfs-kernel-server openssh-server squashfs-tools ethtool net-tools epoptes ipxe memtest86
```

Create chroot area

```
sudo mkdir -p /srv/ltsp
cd /srv/ltsp
sudo apt install debootstrap
sudo debootstrap bullseye x86_64
sudo chroot x86_64
apt install --no-install-recommends linux-image-amd64 initramfs-tools
passwd
exit
```

Config exports

1. Config export

   In /etc/exports of the main system

   ```
   /srv/ltsp/x86_64        *(rw,async,crossmnt,no_subtree_check,no_root_squash,insecure)
   ```

2. Export via NFS

   ```
   sudo exportfs -ra
   ```

Create chroot script (optional)

```
#!/bin/bash
CHROOT_PATH=/srv/ltsp/x86_64
sudo mount -o bind /proc ${CHROOT_PATH}/proc
sudo mount -o bind /dev ${CHROOT_PATH}/dev
sudo mount -o bind /dev/pts ${CHROOT_PATH}/dev/pts
sudo mount -o bind /sys ${CHROOT_PATH}/sys
sudo chroot ${CHROOT_PATH} /bin/bash
CHROOT_PID=$!
wait $CHROOT_PID
sudo umount $CHROOT_PATH/sys
sudo umount $CHROOT_PATH/dev/pts
sudo umount $CHROOT_PATH/dev
sudo umount $CHROOT_PATH/proc
```

Chroot & install required packages

1. Then run the script

   ```
   bash ./ltsp-chroot-bind
   ```

2. Install LTSP on chroot area

   ```
   apt install --install-recommends ltsp
   ```

3. Install xfce4

   ```
   apt install xfce4
   ```

4. Exit chroot

   ```
   exit
   ```

Build image (chroot)

Using zstd build the chroot into an SquashFS image

```
sudo ltsp -m '-comp zstd' image x86_64
```

Build kernels

Copy kernels to tftpboot area

```
sudo ltsp kernel
```

Configure iPXE boot menu

Setup default iPXE boot menus for LTSP

```
sudo ltsp ipxe
```

NFS server config

Setup LTSP shares over NFS (/srv/ltsp)

```
sudo ltsp nfs
```

Generate ltsp.img

Create initrd with passwords, users, groups, etc

```
sudo ltsp initrd
```

Setup Server variable for LTSP

1. Create

```
sudo install -m 0660 -g sudo /usr/share/ltsp/common/ltsp/ltsp.conf /etc/ltsp/ltsp.conf
```

2. Set line under [common] section

```
SERVER=<LTSP VM IP>
```

3. Run ltsp initrd

```
ltsp initrd
```

Setup ltsp.ipxe

Set srv within /etc/ltsp/tftp/ltsp/ltsp.ipxe on vultr

```
set srv <LTSP VM IP>
```

This is important for NFS to work

Open up firewall

On your LTSP VM allow your Office to connect

```
sudo ufw allow from <Office External IP>
```

Start http server

```
cd /srv/tftp
sudo python3 -m http.server
```

TEST

At this stage you should be able to boot

Extras

Automate webserver

1. Add to /etc/ltsp/ltsp.conf

```
[server]
# Use http instead of tftp:
POST_SERVICE_HTTP="python3 -m http.server --directory /srv/tftp &"
```

2. Restart ltsp

```
systemctl restart ltsp
```

Nginx

Instead of usnig python webserver, use Nginx.

Add to /opt/nginx/conf/nginx.conf

```
# -----------------------------------------------------------------------------
# LTSP iPXE boot
# -----------------------------------------------------------------------------
server {
  listen 80;
  server_name ltsp.yourdomain.com.au;
  root /srv/tftp;

  location / {
    index index.html;
    autoindex on;
  }
}
```

Automatic login

- Works with XFCE4 + LightDM
- Base64 your password

```
echo  'password' | base64
```

- Add to /etc/ltsp/ltsp.conf

```
[80:ee:73:00:00:00]
HOSTNAME=ltsp144
AUTOLOGIN=john
RELOGIN=0
PASSWORDS_LTSP144="john/S2FonwXlUgak"
```

- Update

```
sudo ltsp initrd
```

Boot Ubuntu

https://github.com/ltsp/ltsp/wiki/Non-LTSP-iPXE-entries

Copy ipxe template to custom

```
sudo cp /usr/share/ltsp/server/ipxe/ltsp.ipxe /usr/share/ltsp/server/ipxe/ltsp-en.ipxe
```

Download

```
sudo mkdir /srv/ltsp/isos;cd /srv/ltsp/isos
sudo wget https://releases.ubuntu.com/21.10/ubuntu-21.10-desktop-amd64.iso
```

Mount

```
cd /srv/ltsp
sudo mkdir ubuntu
sudo mount -o loop,ro isos/ubuntu-21.10-desktop-amd64.iso ubuntu
```

Add to /etc/fstab (optional)

1. Add to /etc/fstab

```
# Mount loopback devices for iPXE
/srv/ltsp/isos/ubuntu-21.10-desktop-amd64.iso   /srv/ltsp/ubuntu        iso9660 loop,ro 0 0
```

2. Umount

```
sudo umount -l /srv/ltsp/ubuntu
```

3. The mount

```
mount -a
```

Update kernels

```
sudo ltsp kernel ubuntu
```

Add to iPXE menu /usr/share/ltsp/server/ipxe/ltsp-en.ipxe

Menu section just below 'Other options'

```
item --key u ubuntu              Ubuntu 21.10 64bit ISO
```

Below :ltsp section

```
:ubuntu
set cmdline root=/dev/nfs nfsroot=${srv}:/srv/ltsp/${img} netboot=nfs boot=casper ip=dhcp
kernel /ltsp/${img}/vmlinuz initrd=ltsp.img initrd=initrd.img ${cmdline}
initrd /ltsp/${img}/initrd.img
boot || goto failed
```

Update ipxe

```
ltsp ipxe
```

Boot thinclient

NOTICED: Everything seemed to work. Took 3:41mins to boot on an old thinclient (2012, 2GB RAM, 1.1Ghz)

Boot System Rescue CD

Ref: Here config for SystemRescueCD 8 | FOG Project

Download

```
sudo mkdir /srv/ltsp/isos;cd /srv/ltsp/isos
sudo wget https://sourceforge.net/projects/systemrescuecd/files/sysresccd-x86/9.01/systemrescue-9.01-amd64.iso
```

Add to /etc/fstab (optional)

1. Make dir

```
sudo mkdir /srv/ltsp/systemrescue
```

2. Add to /etc/fstab

```
# Mount loopback devices for iPXE
/srv/ltsp/isos/systemrescue-9.01-amd64.iso      /srv/tftp/ltsp/systemrescue     iso9660 loop,ro 0 0
```

3. Umount

```
sudo umount -l /srv/ltsp/systemrescue
```

4. The mount

```
mount -a
```

Add to iPXE menu /usr/share/ltsp/server/ipxe/ltsp-en.ipxe

Menu section just below 'Other options'

```
item --key r systemrescue              System Rescue 9.01 AMD64
```

Below :ltsp section

```
:systemrescue
kernel /ltsp/${img}/sysresccd/boot/x86_64/vmlinuz archisobasedir=sysresccd ip=dhcp \
archiso_http_srv=http://ltsp.example.com.au/ltsp/systemrescue/ checksum
initrd /ltsp/${img}/sysresccd/boot/intel_ucode.img
initrd /ltsp/${img}/sysresccd/boot/amd_ucode.img
initrd /ltsp/${img}/sysresccd/boot/x86_64/sysresccd.img
boot || goto failed
```

NOTE: kernel line is wrapped

Update ipxe

```
ltsp ipxe
```

Boot thinclient

NOTICED: Couldn't start firefox, but terminal and gparted worked.

Demo

Booting

Use PIKVM #6 for demo

Building iPXE

MeshCentral -> debian11-ipxe-buildweb

4. Thinlinc (basic)

https://www.cendio.com/thinlinc/docs/install

This is used to boot into a VM (proper thin client) using the CPU & Memory of the server.

Chroot

```
cd /srv/ltsp
bash ./ltsp-chroot-bind
```

Install

1. Download

   https://www.cendio.com/thinlinc/download

   ```
   wget https://www.cendio.com/downloads/clients/thinlinc-client_4.14.0-2324_amd64.deb
   ```

2. Install

   ```
   sudo dpkg -i thinlinc-client_4.14.0-2324_amd64.deb
   ```

3. Exit chroot

   ```
   exit
   ```

Build image

```
sudo ltsp -m '-comp zstd' image x86_64
```

Test

1. Boot thinclient
2. Start thinlinc
3. Connect to your thinlinc server

References

LTSP

- https://ltsp.org/docs/installation/
- https://github.com/ltsp/ltsp/wiki/chroots

pfSense

- http://blog.smartcore.net.au/smartos-ipxe-boot-with-pfsense/ Great diagrams and explanations.
- https://www.pfsense.org/

iPXE build

- https://gitlab.com/fortnebula/ipxe-web/-/blob/master/README.md
- https://github.com/xbgmsharp/ipxe-buildweb
- https://github.com/xbgmsharp/ipxe-buildweb/blob/master/Dockerfile
- https://ipxe.org/embed

Thinlinc

- https://www.cendio.com/thinlinc/docs/install

Questions

Email	map7@fastmail.com
Twitter	@map7
Github	github: map7