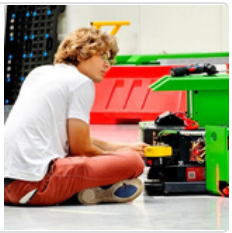
Kvm PXE boot: Save storage, Automate VM rollout

Check GitHub for helpful DevOps tools:

Michael Robotics

Hi, I'm Michal. I'm a Robotics Engineer and DevOps enthusiast. My mission is to create skill-learning platform that combats information overload by adhering to the set of principles: simplify, prioritize, and execute.

https://github.com/MichaelRobotics



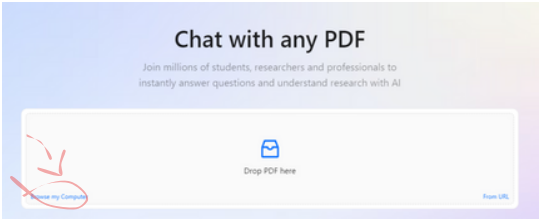
Ask Personal AI Document assistant to learn interactively (FASTER)!

1. Download PDF
2. Go to website
3. Browse file
4. Chat with Document

1 https://github.com/MichaelRobotics/DevOpsTools/blob/main/KVMPxe.pdf

Click there to go to ChatPdf website 2

3



Ask questions about document! 4

Complety new to VM’s and Networking?

If you are completely new to this topic, using a document assistant to understand the many definitions can be helpful. However, the best way to start is by watching this video, which I believe provides the best explanation for beginners starting their journey with Linux and virtualization:

QEMU/KVM for absolute beginners

On this episode of Veronica Explains, I explain the absolute basics of hypervisors generally, KVM specifically, and virt-manager graphically.

https://youtu.be/VeronicaExplains

Essential for this PDF is a thorough knowledge of networking. I highly recommend the HTB platform's networking module, which offers extensive information to help build a comprehensive understanding.

HTB - Your Cyber Performance Center

We provide a human-first platform creating and maintaining high performing cybersecurity individuals and organizations.

https://www.hackthebox.com/

What is KVM PXE boot?

PXE technology allows a computer with a PXE-enabled network interface and firmware to bootstrap from a network server, enabling fully automated OS installations on virtual machines in KVM through a network installation server configured with PXE, DHCP, TFTP or HTTP and NFS.

How KVM PXe boot works?

When using PXE for automated OS installation, the client machine sends a DHCPDISCOVER broadcast during boot, prompting the DHCP server to respond with an IP address and the PXE server location. The client then downloads a boot file from the TFTP or HTTP server, which retrieves files, allowing the OS installer to start and boot the client and pull ISO file from NFS.

KVM PXE boot: Why and When

PXE boot is ideal for large-scale OS deployments, system imaging, and diskless environments, allowing centralized management and automation for efficient setup, updates, and recovery. It is also commonly used in disaster recovery, software testing, and data centers for remote OS installations and reprovisioning without manual intervention.

A typical use case for PXE boot is automating the deployment of a standardized OS image across multiple computers in environments like corporate offices or data centers without requiring physical media.

System Requirements

- 8 gb ram
- 50 free gb storage
- ubuntu 22.04
- firmware must enable PXE support
- network interface supports PXE

If you want to install it on a different Linux distro, ask in the comments and I will write an Ansible playbook or bash script.

Kvm PXE boot: Main components & packages

- libvirt-daemon - runs virtualization in background
- qemu-kvm – An opensource emulator and virtualization package that provides hardware emulation.
- virt-manager – A Qt-based graphical interface for managing virtual machines via the libvirt daemon.
- libvirt-daemon-system – A package that provides configuration files required to run the libvirt daemon.
- virtinst – A set of command-line utilities for provisioning and modifying virtual machines.
- libvirt-clients – A set of client-side libraries and APIs for managing and controlling virtual machines & hypervisors from the command line.
- bridge-utils – A set of tools for creating and managing bridge devices.
- TFTP or HTTP Server – Provides the boot images to start the installer, with the command-line options.
- DHCP server – Provides initial client network configurations, and the location of TFTP server with a usable boot image
- NFS server - to provide live OS image

Kvm PXE boot: Install KVM & Setup KVM network with NAT & Setup server VM network

1) Install KVM

I've made a post with PDF KVM installation.

Kvm: Manage VM's like a PRO

Check GitHub for usefull DevOps tools

https://github.com/MichaelRobotics/DevOpsTools/blob/main/KVMInstall.pdf

2) Setup KVM Network

create KVM network configuration:

```
$ sudo nano /etc/libvirt/qemu/networks/br-pxe-net.xml
```

and paste:

```
<network>
  <name>br-pxe</name>
  <forward mode='nat'>
    <nat>
      <port start='1024' end='65535'/>
    </nat>
  </forward>
  <bridge name='br-pxe' stp='on' delay='0'/>
  <ip address='192.168.177.1' netmask='255.255.255.0'>
  </ip>
</network>
```

Create a virtual network using this file file created; modify if need be:

```
$ sudo virsh net-define --file br-pxe-net.xml
```

Enable automatic starting on the bridge created.

```
$ sudo virsh net-autostart br-pxe
```

Then ensure the bridge interface is online

```
$ sudo virsh net-start br-pxe
```

Confirm that the bridge is available and active:

```
$ sudo virsh net-list
```

Terminal should show:

```
laptopdev@laptopdev2:/etc/libvirt/qemu/networks$ sudo virsh net-list
[sudo] password for laptopdev:
 Name      State    Autostart   Persistent
--------------------------------------------
 br-pxe    active   yes         yes
 default   active   yes         yes
```

Start the interface:

```
$ sudo nmcli conn up br-pxe
```

Check your interface connected to Internet (mine is enp0s31f6). Allow nating with your KVM network using IPtables firewall:

```
$ sudo iptables -t nat -A POSTROUTING -o enp0s31f6 -s 192.168.177.1/24 -j MASQUERADE
```

3) Setup KVM Network

Follow my KVMInstall PDF, before this step!

```
$ virt-install \
 --name ubuntu22.04-VM \
 --memory 10240 \
 --vcpus 4 \
 --bridge=br-pxe \
 --disk path=/media/qemu/ubuntu-vm.qcow2,size=30 \
 --cdrom /media/isos/ubuntu-24.04-desktop-amd64.iso \
```

Follow installation steps, then check ethernet connection on VM:

```
 $ sudo nmcli conn show
```

```
vb2@vb2-Standard-PC-Q35-ICH9-2009:~$ sudo nmcli conn show
[sudo] password for vb2:
NAME            UUID                                  TYPE      DEVICE
netplan-enp1s0  cac41fbe-bc18-3d87-bba7-af2af7f8ffab  ethernet  enp1s0
lo              a5e7da02-b8e7-490a-9af9-064660de81fc  loopback  lo
vb2@vb2-Standard-PC-Q35-ICH9-2009:~$
```

Attach available ethernet connection to KVM network:

```
sudo nmcli con mod netplan-enp1s0 \
  ipv4.method manual \
  ipv4.address 192.168.177.2/24 \
  ipv4.gateway 192.168.177.1 \
  ipv4.dns 192.168.177.1 \
  connection.autoconnect yes
```

Check if interface is UP and have ip address

```
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
group default qlen 1000
    link/ether 52:54:00:20:0d:e0 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::5054:ff:fe20:de0/64 scope link
       valid_lft forever preferred_lft forever
```

Kvm PXE boot: Configure PXE boot env

1) Install PXE Linux

```
$ sudo apt install syslinux pxelinux
```

2) Copy boot files

```
sudo cp /usr/lib/PXELINUX/pxelinux.0 /srv/tftp
sudo cp /usr/lib/syslinux/modules/bios/ldlinux.c32 /srv/tftp


# to display the menu:
sudo cp /usr/lib/syslinux/modules/bios/menu.c32 /srv/tftp
sudo cp /usr/lib/syslinux/modules/bios/libutil.c32 /srv/tftp
```

3) Create boot menu configuration file

```
sudo mkdir /srv/tftp/pxelinux.cfg
sudo nano /srv/tftp/pxelinux.cfg/default
```

4) Create HTTPS boot and TFTP boot configurations

Kvm PXE boot: Create TFTP, HTTP, NFS and DHCP server

1) Access the created VM, if it hasn't been entered already

2) Configure TFTP

Install tftp

```
$ sudo apt install tftpd-hpa
```

Modify **/etc/default/tftpd-hpa** add option --verbose

```
TFTP_OPTIONS="--secure --verbose"
```

restart server

```
$ sudo service tftpd-hpa restart
```

3) Configure HTTP

Install lighttpd

```
$ sudo apt install lighttpd
```

edit server.document-root parameter in /etc/lighttpd/lighttpd.conf and paste /srv/tftp

```
server.document-root        = "/srv/tftp"
server.upload-dirs          = ( "/var/cache/lighttpd/uploads" )
server.errorlog             = "/var/log/lighttpd/error.log"
server.pid-file             = "/run/lighttpd.pid"
server.username             = "www-data"
server.groupname            = "www-data"
server.port                 = 80

# strict parsing and normalization of URL for consistency and security
# https://wiki.lighttpd.net/Server_http-parseoptsDetails
File Name to Write: /etc/lighttpd/lighttpd.conf
^G Help        M-D DOS Format    M-A Append        M-B Backup File
^C Cancel      M-M Mac Format    M-P Prepend       ^T Browse
```

3) Configure DHCP server

Install DHCP server

```
$ sudo apt install isc-dhcp-server
```

Paste commands into: **/etc/dhcp/dhcpd.conf**

```
subnet 192.168.177.0 netmask 255.255.255.0 {
 range 192.168.177.100 192.168.177.199
  option routers 192.168.177.2;

  # option 66
  option tftp-server-name "192.168.177.2";
  # option 67
  option bootfile-name "pxelinux.0";
}
```

restart DHCP server

```
$ sudo service isc-dhcp-server restart
```

4) Configure NFS server

Install NFS

```
$  sudo apt install nfs-kernel-server
```

Configure NFS server file **/etc/default/nfs-kernel-server** as is seen on image:

Configure Export filesystem to NFS client, modify file **/etc/exports** and add:

```
$ sudo service isc-dhcp-server restart
```

Kvm PXE boot: Download and Mount ISO image

1) Create folder for an image

```
$ sudo mkdir /srv/tftp/ubuntu
```

2) Download an image

```
$ curl -O https://releases.ubuntu.com/jammy/ubuntu-22.04.4-desktop-amd64.iso
```

3) Mount image and copy ubuntu files

```
sudo mount -o ro,loop ubuntu-22.04.4-desktop-amd64.iso /mnt
sudo cp -r /mnt/. ubuntu/
```

4) export files to nfs server

```
sudo exportfs -av
sudo systemctl restart nfs-kernel-server
```

Kvm PXE boot: Create a Virtual Machine using PXE Boot CLI (Like SysAdmin)

1) Start VM using CLI

Remember to follow my KVM install tutorial!

```
sudo virt-install \
  --name pxe-vm \
  --ram 4048 \
  --vcpus 2 \
  --disk path=/media/qemu/ubuntu-vm.qcow3,size=25 \
  --network=bridge:br-pxe \
  --pxe \
  --os-type linux \
  --os-variant ubuntu22.04
```

Kvm PXE boot: Create a Virtual Machine using PXE Boot GUI

1) Follow guide from this great blog:

Automated KVM Deployment: PXE & Kickstart Tutorial
By following this article to the end, you should be able to configure and use PXE and Kickstart server tools for Virtual Machine provisioning on KVM

https://computingforgeeks.com/install-virtual-machines-on-kvm-using-pxe-and-kickstart/

Navigate to: **Step 10 – Create a Virtual Machine using PXE Boot**

Common troubleshooting

1) KVM VM creation errors

Make sure that you followed all the steps in my previous KVM installation post. Additionally, check the physical connections to ensure cables haven't come loose from the PC. Check path where you locate your VM's filesystems, remember about limited amount of RAM and memory you have available on your PC.

2) DHCP, TFTP, HTTP, NFS services have "failed" status

How to check services:

```
$ sudo systemctl status isc-dhcp-server
```

```
$ sudo systemctl status tftpd-hpa
```

```
$ sudo systemctl status nfs-kernel-server
```

```
$ sudo systemctl statusstatus lighttpd
```

How to check logs:

```
sudo journalctl -u isc-dhcp-server
```

Check all logs analogically.

3) Cannot mount the ISO image

Check if ISO downloaded properly - does it have right size?

4) Check the nmcli man page

5) If everything is a complete mess

Remove the bridge and revert the configuration to its previous state.

Kvm PXE boot: How to remove

1) Stop and Disable the Services:

```
sudo systemctl stop tftpd-hpa
sudo systemctl stop lighttpd
sudo systemctl stop nfs-server
sudo systemctl stop isc-dhcp-server
```

```
sudo systemctl disable tftpd-hpa
sudo systemctl disable lighttpd
sudo systemctl disable nfs-server
sudo systemctl disable isc-dhcp-server
```

2) Remove the packages

```
sudo apt remove --purge lighttpd
sudo apt remove --purge nfs-kernel-server
sudo apt remove --purge isc-dhcqp-server
sudo apt remove --purge syslinux pxelinux
sudo apt remove --purge tftpd-hpa
```

3) Remove directories

```
sudo rm -rf /srv/tftp
```

Learn more about KVM

Check RedHat, they have great docs

Virtualization Deployment and Administration Guide

Installing, configuring, and managing virtual machines on a RHEL physical machine

https://docs.redhat.com/en

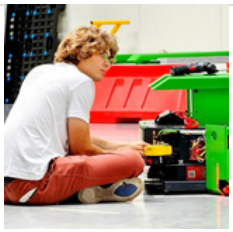
Share, comment, DM and check GitHub for scripts & playbooks created to automate process.

Check my GitHub

Michael Robotics

Hi, I'm Michal. I'm a Robotics Engineer and DevOps enthusiast. My mission is to create skill-learning platform that combats skill information overload by adhering to the set of principles: simplify, prioritize, and execute.

https://github.com/MichaelRobotics



PS.

If you need a playbook or bash script to manage KVM on a specific Linux distribution, feel free to ask me in the comments or send a direct message!