LFCS Crash Course

2023 Edition

Sander van Vugt

Pearson

Notes

- For additional information, study "Linux Foundation Certified Systems Administrator (LFCS), 3rd edition"
- The study materials provided by Linux Foundation cover about 70% or required exam topics
- The exam machines are running Ubuntu Server 22.04
- On the exam, no Ubuntu specifics have been observed: you should be able to pass based on knowledge of other distributions
- The mission of this course is to cover all topics

Course Expectations

- This course assumes you have at least intermediate knowledge of Linux Fundamentals
- If you feel week on Linux Fundamentals, attend my "Linux Fundamentals Bootcamp"
- Alternatively, watch my "Linux Fundamentals" recorded course, available on this platform
- Your instructor will not have time to answer questions related to Linux Fundamentals

Lab Requirements

- Recommended: set up a virtual machine that runs Ubuntu Server LTS 22.04
- Alternative: Use the O'Reilly Linux Sandbox (expires after 60 minutes)

Generic Skills

Generic Skills

- sudo apt install apt-file
- sudo apt-file update
- sudo apt-file search myfile
- sudo mandb; man -k anything

Lab1: Finding Packages (5 minutes)

- Find the package that contains the seinfo file

User Management

User Management

- all require sudo
- useradd -m isabelle
- groupadd profs
- groupadd staff
- useradd -m -G profs isabelle
- useradd -m -s /usr/bin/nologin marcha
- usermod -aG staff isabelle
- usermod -g staff isabelle
- passwd isabelle

Lab 2: Managing Users (7 minutes)

- Create the groups sales with GID 1024 and account with GID 1025
- Create user anna and makes sure she is a member of the group sales as a secondary group. She should also have a home directory
- Change primary group membership for anna to the group sales
- Create user linda and ensure she cannot use an interactive shell
- Set passwords for all users to "mypassword"

Permissions

- Basic File access permissions define what you can do where on directories and files:
  - read (4): allows reading on files and listing on directories
  - write (2): allows modifying files and deleting or adding in directories
  - execute (1): allows running programs and should always come on directories if a user has read on the directory
- First take of ownership before setting permissions (chown, chgrp)
- Next, use chmod to set permissions on files and directories
  - chmod +x myscript
  - chmod 750 /my/directory

Resource restrictions

Understanding Resource Restrictions

- **ulimit** is the legacy way to implement resource restrictions
- Modern Linux offers Cgroups through systemd
- Using Cgroups is recommended
- On the exam, make sure you know how to work with **ulimit**
- Find the file **limits.conf**, have a look at it and you'll know what to do

Lab 3: Setting Resource Restrictions (5 min)

- Configure a soft limit such that user lisa cannot start more than 4 processes and set the hard limit to 8

Storage Management

Essential tools

- du
- df

Lab 4: Monitoring Storage (5 min)

- Create a list of all storage devices that currently are using less than 90% of their disk space and write the names of these devices to the file /tmp/storage-use.txt

Partition Management

Procedure overview

- Use **lsblk** to identify disk device and partition names, and check unallocated disk space
- Use **fdisk** to create the new partition
  - **p** will print the current layout
  - **n** allows to create a new partition
  - **w** write/quit fdisk
- Use **mkfs.xxx** to create a FS on the new partition
- **mount /dev/xxx /somedir** to mount
- Make the mount persistent in /etc/fstab:
  - **/dev/nvme0n1p4 /data ext4 defaults 0 0**

LVM Management

Procedure Overview

1. Create a partition of type LVM or provide a complete disk
2. pvcreate to mark the block device as a PV
3. vgcreate to create the VG based on one or more PVs
4. lvcreate to create an LV from the VG
5. mkfs.xxx /dev/vgname/lvname to format it
6. mount it somewhere
7. don't forget about fstab

Essential Tools

- lvs
- vgs
- lvresize -r

Resizing LVM overview

1. Check for available free in **vgs** output
2. If no free in **vgs** output, use **vgextend** to add a block device
3. Use **lvextend -r -l +nn%FREE /dev/vgname/lvname**
4. Use **lvs** to verify the *logical volume* has been resized
5. Use **df -h** to verify the *filesystem* has been resized
6. In case the filesystem was not resized, use **resize2fs** to resize later

Exercise: add 1GB to the previously created LVM logical volume, and make sure you see it in the filesystem

Lab 5: Configuring LVM (15 minutes)

- Use the **dd** and **losetup** utilities to create a loop device with a size of 1GiB
- Create an LVM volume group with the name vgdata that uses 800 MiB of this loop device
- Create an LVM logical volume with the name lvdata with a size of 600 MiB
- Format the logical volume with the Ext4 file system and mount it persistently on the directory /data
- Resize the logical volume to grow to a size of 750 MiB. The Ext4 file system should also be grown and you're not allowed to unmount the logical volume

SSH Service Management

Essential Configuration

- The Generic SSH config file is /etc/ssh/sshd_config
- User specific options can be stored in ~/.ssh/config
- Secure SSH implementations have passwords disabled by default
  - KbdInteractiveAuthentication no
  - PasswordAuthentication no

Lab 6: Configuring SSH (8 min)

- Create an SSH configuration that allows user anna to log in with a password, but not user linda
- You don't need an external server to perform this lab, just configure localhost accordingly

Analyzing Performance

Essential Tools

- iostat
- top
- iotop
- kill
- lsof

Lab 7: Managing Storage Performance (5 min)

- Find the busiest storage device
- Stop the process that causes the highest write load on that storage device
- Write the name of that process to the file /tmp/storage.txt

Using find

Essential Tools

- find /mydir -name "*.txt" -size +5m -exec rm {} \;

Lab 8: Finding Files (4 min)

- Find all files that are owned by user anna and copy them to the directory /root/anna/. Use one single commandline to perform this task

Running a Libvirt Virtual Machine

Starting a VM from a disk image

- grep -e vmx -e svm /proc/cpuinfo
- sudo apt install kvm* qemu*
- sudo apt install libvirt* virtinst
- sudo systemctl enable --now libvirtd
- wget https://cloud-images.ubuntu.com/jammy/current/jammy-server-cloudimg-amd64-disk-kvm.img
- virsh net-list --all
- virsh net-start default
- virt-install --disk /tmp/jammy-server-cloudimg-amd64-disk-kvm.img --memory 512 --osinfo detect=on,require=off --name myvm --boot hd
- virsh list

Lab 9: Starting a VM from a Disk Image (15 min)

- Fetch a generic Ubuntu Cloud image from https://cloud-images.ubuntu.com/jammy/current/jammy-server-cloudimg-amd64-disk-kvm.img
- Run this image with the name "myvm" and 256 MiB RAM

Managing Cron Jobs

Essential Tools

- su - lisa
- crontab -e
- logger
- journalctl
- /etc/rsyslog.conf

Lab 10: Managing Scheduled Tasks (5 min)

- Configure a scheduled task that runs as user anna from Monday through Friday at 5PM. The task should write the message "I'm going home" to the logging system that is in use on your server

Compiling Software from Source

Essential Tools

- apt install autoconf make gcc
- gcc
- ./configure
- make
- make install

Lab 11: Compiling from Source (10 min)

- Clone the course Git repository at https://github.com/sandervanvugt/lfcs (original at https://github.com/ewxrjk/fingerd/)
- Compile the source code that you find in the finger directory

Managing NFS Servers

Key Components

- systemctl status nfs-server
- /etc/exports
- showmount -e

Lab 12: Configuring an NFS Server

- Create an NFS server that offers the following exports
  - server1 should have full access to /media/server1
  - server2 should have read-only access to /media/server2
  - server1 and server2 should have full access to /tmp/servers
- Make sure to create the shared directories
- Verify that it works

Managing NTP Servers

Key Tools

- systemctl status ntpd
- /etc/ntpd.conf
- ntpdate -q ntp.server.pool.org

Lab 13: Configuring Time (5 minutes)

- Configure the NTP time service to synchronize time with 5 servers in pool.ntp.org
- Verify that your computer is synchronizing correctly

Troubleshooting Systemd Services

Essential Tools

- systemctl status
- journalctl

Lab 14: Verifying apache2 working

- Run the script run-apache2.sh from the course Git repository at https://github.com/sandervanvugt/lfcs
- Do NOT open the script to read its contents!
- This script should install and run the apache2 web service
- Verify that it is working correctly

Analyzing TLS Files

Essentials

- About public/private keys, certificates and CA's
- openssl x509 -in mycert.pem -noout -text

Demo: Creating self-signed Certificates

- Creating the CA: "Creating a Self-signed RootCert"
  - **mkdir ~/openssl**
  - **openssl genrsa -des3 -out myCA.key 2048**
  - **openssl req -x509 -new -nodes -key myCA.key -sha256 -days 3650 -out myCA.pem**
- Creating the certificate: "Create a private key and generate a certificate request from it"
  - **openssl genrsa -out tls.key 2048**
  - **openssl req -new -key tls.key -out tls.csr** # make sure the CN matches the DNS name of route (created later) which is linginx1-myproject.apps-crc.testing
- Self-signing the certificate: "Sign a certifcate request"
  - **openssl x509 -req -in tls.csr -CA myCA.pem -CAkey myCA.key -CAcreateserial -out tls.crt -days 1650 -sha256**

Lab 15: Using TLS Certificates (10 minutes)

- Fetch the file tls.crt from the course Git repository at https://github.com/sandervanvugt/lfcs
- Print the common name in the certificate and write it to the file /tmp/tlscert.txt
- Copy the certificate to the directory on the server where it will be available for generic use

Firewall Port Forwarding

Required Knowledge

- sudo apt install firewalld
- man 5 firewalld.richlanguage
- firewalld add-rich-rule "xxxx"

Lab 16: Configuring Port Forwarding (10 min)

- Configure port forwarding to meet the following requirements
- All traffic addressed to port 80 and coming from the network 172.19.0.0/16 should be forwarded to port 8081 on the localhost
- All other traffic addressed to port 80 should be forwarded to port 8082 on the localhost

NTS: Solution

- firewall-cmd --permanent --zone=public --add-rich-rule='rule family="ipv4" source address="172.19.0.0/16" forward-port to-port="8081" to-addr="127.0.0.1" protocol="tcp" port="80"'
- firewall-cmd --permanent --zone=public --add-rich-rule='rule family="ipv4" forward-port to-port="8088" to-addr="127.0.0.1" protocol="tcp" port="80"'
- firewall-cmd --reload

Docker

Key Skills

- docker build -t myapp:latest
- man docker-run
- docker run -d -p -m 512 myapp

Lab 17: Managing Containers (15 minutes)

- Use the Dockerfile in the course Git repository to build an image with the name myapp:latest
- Start a container based on this image
- Also start a container based on the nginx image from the Docker Hub registry. Ensure that it runs in the background. Expose port 80 in the container to port 8080 on the host that runs the container. Limit the memory this container can use to 256 MiB and ensure it starts when the host computer is starting

Git Repositories

Git essentials

- git add
- git commit -m mymessage
- git push

Tweaking Kernel Parameters

Tweaking Essentials

- /proc/sys
- sysctl -a
- net.ipv4.ip_forward
- /etc/sysctl.conf

Lab 18: Tweaking kernel parameters (5 min)

- Modify kernel settings persistently such that:
  - IP forwarding is enabled
  - Swappiness is set to a value of 60