CIS245 - LINUX ADMINISTRATION

HOLLY MONTALVO

CONTENTS

ABSTRACT

This document will attempt to outline the process taken to configure Acme Corporation's Buizel and Lycanroc[1] servers. This will include the setup, installed applications, links to scripts utilized, and rationale for all procedures taken. Note that, until the estimated completion date of 6 May 2020, this document should be considered a living document, and as such, its contents may be changed at any time, including portions that may seem to be complete.

[1] *I'm certain Nintendo won't get mad about the names of these servers. I mean, they're not aggressive with cease and desists, right? Right?!?*

1 INSTALLATION

1.1 CentOS Linux 8

1.1.1 *Choosing CentOS Version*

Upon downloading CentOS, you are given an option to download CentOS Linux or CentOS Stream. While for personal projects I would be interested in using CentOS Stream to potentially gain access to newer sources, for the purposes of running and administering a company server, the only sensible option is CentOS Linux. CentOS Stream is a rolling distribution, designed to help contribute to future releases of Red Hat Enterprise Linux - and rolling distributions aren't always the most stable, which means a higher likelihood of downtime that could have been avoided. In regards to OS version, I will be rolling with CentOS 8. CentOS 8 would give us full updates until May 2024, and maintenance updates until May 31, 2029. On the other hand, we would no longer receive full updates for CentOS 7 starting Q4 of 2020, and would no longer receive maintenance updates starting June 30th, 2024. While the rollout for CentOS 8 has been claimed to not exactly be the best, I feel like this additional challenge shouldn't cause too much of a problem.

1.1.2 *Booting into Anaconda*

The CentOS 8 installer is known as Anaconda, which starts up with the install media. The first screen is relatively straightforward - as long as you speak English and live in the United States, it's probably best to stick with the defaults here. You are then taken to a page that may seem a tad overwhelming at first – but fear not! Take it step by step and it's relatively simple.

1.1.3 *Anaconda - Localization*

The Localization column is the most straightforward - it defaults to English (US) for the keyboard and Americas/New York for the timezone. If either of these need to be changed (whether you live in another timezone or you use a different keyboard layout), these can be changed by clicking on them and making the appropriate changes.

1.1.4 *Anaconda - System*

Under the System column, select "Network & Host Name." Set your desired system name under Host Name. I chose buizel.server.acme for this server. If you have a DHCP server running on the network, you can go ahead and enable the Ethernet at this point. (If you've ever connected a device to the Internet and didn't need to give it an IP address, subnet, gateway, and DNS server addresses, you have a DHCP server on your network.) Otherwise, select "Configure..." and enter the appropriate information under IPv4 settings (and IPv6, if your network has it enabled.) Below is an example static IP setup with IPv4.

Back at the Summary screen, select "Installation Destination". You can leave everything at this screen at the defaults, but if you know what you're doing, you can also select "Custom" and make whatever adjustments you see fit to the partitioning of the disk. This, however, is beyond the scope of this server install - the defaults will be fine in our case.

1.1.5 *Anaconda - Software*

We'll need to make a change under the Software column. You don't want to install a GUI with the server - it's an unnecessary waste of resources and you'll find yourself wondering why you did it. Click "Software Selection" and select "Server" for the

Base Environment. Don't worry about checking any of the boxes under "Additional software" - we can install these later.

1.1.6 *Anaconda - Begin Installation*

At this point, the installer will allow you to continue to installing the system - press "Begin Installation" to kickstart the installation process. At this point, the installer will begin to install files to disk and you will be given the option to create a root password and create a user. Do both of these - make sure the root password is extremely strong, as this password would give someone complete and total control over the server. Under User Creation, make sure "Make this user administrator" is checked, and set just as strong of a password, as this account would be able to log into the root account with its credentials.

1.1.7 *Finishing the Installation*

At this point, you can pretty much just sit back as the install completes! You'll know it's finished when it says "CentOS Linux is now successfully installed and ready for you to use! Go ahead and reboot to start using it!" Press the blue Reboot button to restart into the new OS. Once you reach this screen, you'll want to login. Then, once logged in, you'll want to update anything that might need updating using the command `sudo yum update`.

1.2 Ubuntu Server 18.04

1.2.1 *Choosing a Linux Distribution*

While Acme Corporation had decided upon CentOS for its first system, it appeared there was still disagreement as to which distribution should be used for the second server. As I was ultimately given responsibility to determine what distribution would be run on the second operating system, I chose Ubuntu Server[2].

I personally find myself most productive when utilizing Debian-based operating systems, and Debian-based systems are relatively simple to set up and operate, as resources and support for it are plentiful. With a personal server for my hobbies, I would choose Ubuntu 19.10 to gain hassle-free access to more recent software releases, or Debian 10 out of personal preference for Debian over Ubuntu, but in this circumstance I chose Ubuntu 18.04.3 LTS. I have a few reasons for this:

1. While more recent software releases are nice to have, stability is more important in a commercial environment.
2. Ubuntu Server 18.04.3 LTS installations are guaranteed support until April 2023, as opposed to 19.10 which only has support until July 2020 (at which point you'd be expected to upgrade to continue receiving support.)
3. Canonical, the company which manages Ubuntu, provides enterprise support by the name of Ubuntu Advantage, as opposed to Debian which doesn't have a similar level of enterprise support. This is a nice option to have, even if it's something that Acme Corporation will never choose to use.

[2] *I can't help but admit I was slightly tempted to choose Gentoo Linux or Linux From Scratch for the sake of a challenge, but I decided against that not only due to their complexity, the lack of real benefit it'd present to Acme Corporation, the amount of time I'd need to wait for everything to compile, and because this document could literally be bound into a book of not insignificant size if I explained the configuration of such a system.*

1.2.2 *Booting into Subiquity*

Install is pretty straightforward to a point. You can download the ISO from this website. Boot to the ISO from the server - the method of doing so may vary from system to system, but there are guides on the Ubuntu website on how to create installable media. You may even be able to supply the ISO directly to a management GUI for the server, depending on your setup. Upon booting up the ISO, you will be greeted to the Subiquity installer - the install GUI for Ubuntu Server. The first two screens are relatively straightforward - if you speak English and use a standard keyboard layout, you can safely skip those screens. If you speak another language, you can select it on the first screen, and if you have a non-standard keyboard, select your keyboard layout from the Variant drop down - or, alternatively, select [Identify keyboard] and the installer will have you press a series of buttons that will narrow down what type of keyboard you have.

1.2.3 *Subiquity - Internet*

If your network has a DHCP server (if you've ever connected a device to the Internet and didn't need to give it an IP address, subnet, gateway, and DNS server addresses, you have a DHCP server on your network), you can hit [Done] on this screen without further configuration. If not, select your network device from the screen, select Edit IPv4, select Manual, and fill in the textboxes with the relevant information. (You should be aware of them - if not, please refer to your network setup or contact the network administrator.) If you have IPv6 connectivity, you may also wish to set that up in a similar manner. A sample configuration for IPv4 can be seen below. If you use an HTTP proxy to access the Internet from your network, you'll need to put it in at the 4th screen (You likely don't.) It's safe to leave the Ubuntu archive mirror at the default for the 5th screen.

1.2.4 *Subiquity - File System*

The seventh screen will ask what you would like for the file system setup. You can leave the defaults here as-is. However, if you know what you're doing, it can be beneficial to manually partition different directories to different sections of the drive. It's a little beyond the scope of the installation we're doing here, however, so just select [Use An Entire Disk]. (Note - you'd also want to do some additional configuration if this isn't the only OS on your hard drive, however... why would you be dual booting a server?)

1.2.5 *Subiquity - Profile and Apps*

At this point, the installer will proceed. Fill in your user details and a server name as appropriate, then proceed to the next screen. When asked, choose to install the OpenSSH server. Don't import an SSH identity at this time - we'll take care of this later. You will then be given a smattering of different applications you may want to install. I personally don't select anything in this step, as I find it better to just install what I need down the line.

1.2.6 *Finishing the Installation*

At this point, the installer's asked everything of you that it needed and will finish the installation on its own. Let it do its thing and install security updates (you'll be downloading and installing said updates either way - might as well get it out of the way now.) Eventually, there will be a button that just says [Reboot] - once you get to this point, select that option, remove the installation media when prompted, and boot up into the newly installed operating system.

Once you get to this screen, log in with the username and password you set up during installation. Once in, run the following commands to ensure your system is fully updated:

```
$ sudo apt update
$ sudo apt upgrade -y
```

2 FILE SHARING

When it comes to performing file transfers to and from the server, you really can't go wrong with `scp` - while it isn't as user friendly as using something like WinSCP, Firezilla, or Cyberduck, it well makes up for it in its functionality.

Using `scp` to perform file transfers is as simple as typing a command with a few arguments. The basic command to upload files is as follows:

```
$ scp [source file] [user]@[host]:[destination file]
```

and the command to download files is:

```
$ scp [user]@[host]:[source file] [destination file]
```

This is the basic command to do what you want. Also, there are a few arguments you may want to add, depending on the situation:

- `-r` to copy entire folders
- `-i [private key]` to use pubkey authentication

So, if I wanted to download, say, the /var/logs/ folder from a server at holly.example.com while logged in as frinkel using a private key, I'd run this command:

```
$ scp -r -i ~/myprivatekey frinkel@holly.example.com:/var/logs/ \
~/Documents/serverlogs/
```

3 INSTALLING APPLICATIONS

3.1 CentOS Linux 8

3.1.1 *Updating CentOS*

Updating with CentOS could only hardly be easier. All it takes is running one command:

```
$ sudo yum update
```

Yum will automatically fetch updates for all the repositories, then check for updates. If there are updates, it'll have you confirm them, and then you'll be on your way to a more up-to-date system.

3.1.2 *Viewing Included Repositories*

Acquiring a list of all of the enabled repositories on a CentOS system is also pretty trivial. It just takes one simple command:

```
[holly@buizel ~]$ yum repolist enabled
Last metadata expiration check: 0:00:29 ago on Fri 06 Mar 2020 03:10:54
PM EST.
repo id                    repo name                             status
```

```
AppStream                 CentOS-8 - AppStream                 5,103
BaseOS                    CentOS-8 - Base                      2,107
extras                    CentOS-8 - Extras                        3
```

You can, alternatively, show a list of only disabled repositories, or all repositories period, by replacing `enabled` with `disabled` or `all`, respectively.

3.1.3 *Installing Applications*

Installations are pretty much just as simple as they are on Ubuntu/Debian.

```
[holly@buizel ~]$ sudo yum install tmux
Last metadata expiration check: 0:01:50 ago on Fri 06 Mar 2020 03:10:36
PM EST.
Dependencies resolved.
=========================================================================
 Package       Architecture    Version           Repository       Size
=========================================================================
Installing:
 tmux          x86_64          2.7-1.el8         BaseOS          317 k

Transaction Summary
=========================================================================
Install  1 Package

Total download size: 317 k
Installed size: 781 k
Is this ok [y/N]: y
Downloading Packages:
tmux-2.7-1.el8.x86_64.rpm              1.3 MB/s | 317 kB     00:00
-------------------------------------------------------------------------
Total                                  692 kB/s | 317 kB     00:00
warning: /var/cache/dnf/BaseOS-929b586ef1f72f69/packages/tmux-2.7-1.el8.
x86_64.rpm: Header V3 RSA/SHA256 Signature, key ID 8483c65d: NOKEY
CentOS-8 - Base                        304 kB/s | 1.6 kB     00:00
Importing GPG key 0x8483C65D:
 Userid     : "CentOS (CentOS Official Signing Key)
 <security@centos.org>"
 Fingerprint: 99DB 70FA E1D7 CE22 7FB6 4882 05B5 55B3 8483 C65D
 From       : /etc/pki/rpm-gpg/RPM-GPG-KEY-centosofficial
Is this ok [y/N]: y
Key imported successfully
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                               1/1
  Installing       : tmux-2.7-1.el8.x86_64                         1/1
  Running scriptlet: tmux-2.7-1.el8.x86_64                         1/1
  Verifying        : tmux-2.7-1.el8.x86_64                         1/1

Installed:
  tmux-2.7-1.el8.x86_64

Complete!
```

Note: You may be asked to confirm a signature, like shown here. Just press Y to continue if that's the case. If you're concerned about security, you can double-check the key "fingerprint" with online sources.

You can also specify multiple applications in the same line:

```
$ sudo yum install emacs vim
Last metadata expiration check: 0:34:54 ago on Tue 25 Feb 2020 08:07:...
Package vim-enhanced-2:8.0.1763-13.el8.x86_64 is already installed.
Dependencies resolved.
*** chart of packages to install here ***
Install  97 Packages

Total download size: 109 M
Installed size: 394 M
Is this ok [y/N]:
```

Note that it will not continue if a package is misspelled or not in any of the repositories.

```
$ sudo yum install cowsay lolcat
Last metadata expiration check: 0:37:38 ago on Tue 25 Feb 2020 08:07:...
No match for argument: cowsay
No match for argument: lolcat
Error: Unable to find a match: cowsay lolcat
```

While not the prettiest, if you can't find a suitable repository for your package for your current version, you can always download the package from an older repository from the Net and install it that way (although this isn't exactly the most lauded solution):

```
$ wget http://download-ib01.fedoraproject.org/pub/epel/7/x86_64/
Packages/c/cowsay-3.04-4.el7.noarch.rpm
*** insert downloading here ***
2020-02-25 08:49:25 (979 KB/s) - ‘cowsay-3.04-4.el7.noarch.rpm’
saved [43144/43144]
$ sudo yum install ./cowsay-3.04-4.el7.noarch.rpm
Last metadata expiration check: 0:11:59 ago on Fri 06 Mar 2020 03:10:36
PM EST.
Dependencies resolved.
***dependency list here***
Transaction Summary
========================================================================
Install  42 Packages

Total size: 12 M
Total download size: 12 M
Installed size: 35 M
Is this ok [y/N]: y
***1 install later...***
Complete!
```

Or, if you can't find a package, you can always do the fun thing and install from source! (On an unrelated note: No wonder CentOS 8 was advised against...)

```
$ sudo yum install ruby
$ wget https://github.com/busyloop/lolcat/archive/master.zip
$ unzip master.zip
$ cd lolcat-master/bin/
$ sudo gem install lolcat
```

(also, as you can tell, the last two were bodges in an attempt for me to make up for the fact that they seemed to have stopped liking fun at CentOS 8 and so it doesn't have cowsay or lolcat on its repositories, and worse, I can't find a repo with a lolcat package.)

3.1.4 *Adding the EPEL Repository*

Some programs require adding the Extra Packages for Enterprise Linux (EPEL) repository to your system. This is the case with `fail2ban`. This can be done with a simple command:

```
$sudo install epel-release
```

and then you can install whatever you'd like that was in the EPEL repository.

```
$ sudo yum install fail2ban
***One install later...***
Complete!
```

3.1.5 *Adding Repositories*

All of CentOS's repositories can be found in the `/etc/yum.repos.d/` folder. Adding repositories can be done in a few ways, such as adding a file in this directory. Here, I will be demonstrating how I prefer to do this using Docker's repository, as it can serve a good example of what you might expect for repository requirements. Docker uses a script located in the `yum-utils` package.

First off, it's worth looking the repository up to see the exact requirements specified for the repository. Docker has such a page for CentOS systems here:
https://docs.docker.com/install/linux/docker-ce/centos/

First, a few programs need to be installed. The `yum-utils` package is of special interest here, as it will provide the `yum-config-manager` program.

```
    $ sudo yum install -y yum-utils \
    device-mapper-persistent-data lvm2
```

Then, it has you use the `yum-config-manager` application to add the `stable` repository to your CentOS install:

```
    $ sudo yum-config-manager --add-repo \
    https://download.docker.com/linux/centos/docker-ce.repo
```

At this point, the repository is installed. When installing an application from the new repository, it'll prompt you to accept the GPG key - do this and you're all good!

```
warning: /var/cache/dnf/(...)/containerd.io-1.2.0-3.el7.x86_64.rpm:
Header V4 RSA/SHA512 Signature, key ID 621e9f35: NOKEY
Docker CE Stable - x86_64               9.0 kB/s | 1.6 kB     00:00
Importing GPG key 0x621E9F35:
 Userid     : "Docker Release (CE rpm) <docker@docker.com>"
 Fingerprint: 060A 61C5 1B55 8A7F 742B 77AA C52F EB6B 621E 9F35
 From       : https://download.docker.com/linux/centos/gpg
Is this ok [y/N]:
```

3.2 Ubuntu Server 18.04

3.2.1 *Viewing Included Repositories*

Viewing repositories can be done by looking inside the `/etc/apt/sources.list` file, as well as within all the files within the `/etc/apt/sources.list.d/` folder. Alternatively, there's a bit of console magic that can list all of the repositories within these files for your viewing pleasure:

```
$ grep -Erh '^deb ' /etc/apt/sources.list* | cut -d ' ' -f 2-
http://us.archive.ubuntu.com/ubuntu bionic main restricted
http://us.archive.ubuntu.com/ubuntu bionic-updates main restricted
http://us.archive.ubuntu.com/ubuntu bionic universe
http://us.archive.ubuntu.com/ubuntu bionic-updates universe
http://us.archive.ubuntu.com/ubuntu bionic multiverse
http://us.archive.ubuntu.com/ubuntu bionic-updates multiverse
http://us.archive.ubuntu.com/ubuntu bionic-backports main restricted
universe multiverse
http://us.archive.ubuntu.com/ubuntu bionic-security main restricted
http://us.archive.ubuntu.com/ubuntu bionic-security universe
http://us.archive.ubuntu.com/ubuntu bionic-security multiverse
http://archive.ubuntu.com/ubuntu/ bionic main restricted
http://archive.ubuntu.com/ubuntu/ bionic-updates main restricted
http://archive.ubuntu.com/ubuntu/ bionic universe
http://archive.ubuntu.com/ubuntu/ bionic-updates universe
http://archive.ubuntu.com/ubuntu/ bionic multiverse
http://archive.ubuntu.com/ubuntu/ bionic-updates multiverse
http://archive.ubuntu.com/ubuntu/ bionic-backports main restricted
universe multiverse
http://security.ubuntu.com/ubuntu/ bionic-security main restricted
http://security.ubuntu.com/ubuntu/ bionic-security universe
http://security.ubuntu.com/ubuntu/ bionic-security multiverse
```

3.2.2 *Updating Ubuntu*

Updating Ubuntu is as simple as running the following commands to update the list of available packages from the repositories on your system, then update the packages:

```
$ sudo apt update
$ sudo apt upgrade -y
```

That's literally all you need to do! (Also, if you remove the `-y` from the second command, it'll also prompt you with the programs it's about to update before doing so.)

3.2.3 *Installing Applications*

Installing applications is relatively straightforward, as long as you have the repositories you need in your system:

```
$ sudo apt install tmux
Reading package lists... Done
Building dependency tree
Reading state information... Done
tmux is already the newest version (2.6-3ubuntu0.2).
tmux set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

(Note: in that example, I already had tmux installed, so apt didn't really do anything (since it didn't *need* to), other than indicate that I specifically asked for that program to be installed.)

You can also specify multiple programs in the same line:

```
$ sudo apt install emacs fail2ban cowsay lolcat vim
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

```
The following additional packages will be installed:
  *** a lot of dependencies here ***
Suggested packages:
  *** a lot of suggested packages here ***
The following NEW packages will be installed:
  *** a lot of dependencies here, plus the requested packages ***
0 upgraded, 145 newly installed, 0 to remove and 0 not upgraded.
Need to get 75.7 MB of archives.
After this operation, 288 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```

Be careful when doing this, though - if you misspell any one of the programs, or one of them isn't included in any one of the repositories on your system, apt will get upset and won't do anything:

```
$ sudo apt install emacs fail2ban cowsay lelcat
Reading package lists... Done
Building dependency tree
Reading state information... Done
E: Unable to locate package lelcat
```

Also, don't forget to run `apt` as root or use `sudo` to run it, or else it'll complain about permissions and demand it be given root access.

```
$ apt install cancer-cure
E: Could not open lock file /var/lib/dpkg/lock-frontend - open (13:
Permission denied)
E: Unable to acquire the dpkg frontend lock
(/var/lib/dpkg/lock-frontend), are you root?
```

3.2.4 *Adding Repositories - PPA*

Ubuntu has a type of repository known as Personal Package Archives, made by third parties that are, admittedly, rather trivial to add to your Ubuntu distribution. Do watch out, though, since they aren't from official sources!

launchpad.net has a list of different PPAs available. Each PPA listing actually informs you of the appropriate commands to run. For example, to install the Sample PPA from kirinnee (also accessible at https://launchpad.net/ kirinnee/+archive/ ubuntu/sample), just run

```
$ sudo add-apt-repository ppa:kirinnee/sample
$ sudo apt-get update
```

You'll be prompted to accept the repository. Do so and you're all set! (Note: This repository wasn't actually added to the Ubuntu server.)

3.2.5 *Adding Repositories*

Adding repositories can be done in a few ways. I will be demonstrating how I prefer to do this using Docker's repository, as it can serve a good example of what you might expect for repository requirements. First off, it's worth looking the repository up to see the exact requirements specified for the repository. Docker has such a page for Ubuntu systems here: https://docs.docker.com/install/linux/docker-ce/ubuntu/

According to the manual, we'll need to install a few applications so it'll work:

```
$ sudo apt-get install apt-transport-https \
    ca-certificates curl gnupg-agent \
    software-properties-common
```

Then we'll need to install and verify the key that Docker uses to sign their installs:

```
$ curl -fsSL https://download.docker.com/linux/debian/gpg | \
sudo apt-key add -
OK
$ sudo apt-key fingerprint 0EBFCD88
pub   rsa4096 2017-02-22 [SCEA]
      9DC8 5822 9FC7 DD38 854A  E2D8 8D81 803C 0EBF CD88
uid           [ unknown] Docker Release (CE deb) <docker@docker.com>
sub   rsa4096 2017-02-22 [S]
```

This appears to match the fingerprint on the Docker website, `9DC8 5822 9FC7 DD38 854A E2D8 8D81 803C 0EBF CD88`, so we should be good. (If you're extremely concerned about security, you can cross-compare this value to what it says on multiple different websites, devices, and internet connections to ensure the value isn't being tampered with.)

Now, we can finally install the repository. The way Docker instructs you to add the repository is with this command:

```
sudo add-apt-repository \
   "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
   $(lsb_release -cs) stable"
```

This adds the repository to the `/etc/apt/sources.list` file and runs `apt get update`. Alternatively, you can take the line it would add to the source.list file and make your own file containing just that in a new file in the `/etc/apt/sources.list.d/` folder:

```
    echo "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
 $(lsb_release -cs) stable" | sudo tee \
 /etc/apt/sources.list.d/docker.list
```

4 USER MANAGEMENT

4.1 Guidelines

The following guidelines are to be followed in regards to user account creation and modification for the Acme Corp Buizel and Lycanroc servers:

4.1.1 *General*

- All user accounts will be generated with the `batchacct.py` script. Such must be done by an administrator, with the script run as sudo.
- Passwords can be changed by users themselves with the `passwd` command.
- The `batchacct.py` script, for every six users in the input file, will assign the first to Admin, second and third to Developer, fourth and fifth to Staff, and the sixth to Temp.

4.1.2 *Usernames*

- For all names, the first letter and the full last word in the name will be used, separated by a period. All caps will be removed.
  Example: Holly Lotor Montalvo -> h.montalvo
- If someone with that same username already exists, a random number between 0-99999 will be generated and appended to the end of the username.
  Example: Hunter Montalvo -> h.montalvo44821

- Users with names that'd generate family unfriendly usernames, or usernames matching common administrative usernames, will have their full first and full last name used instead.
  Example: Seth Lut -> seth.lut, Ashley Dmin -> ashley.dmin
- If any user has a legitimate issue with their username that all prior rules would not resolve, administrative staff may create a suitable username for them similar to the aforementioned naming structure.

4.1.3 *Passwords*

- All users will be given a random twenty four-digit alphanumeric password. This password should be changed as soon as possible.
- Changed passwords must be at least twelve characters long, and must contain at least one capital letter, one lowercase letter, and one number.
- Passwords must be changed at least every 180 days, or 6 months. There must not be any re-use.
- The use of a secure password manager, such as Keepass, Bitwarden, or Lastpass, while not mandatory, is strongly recommended for all users.

4.1.4 *Groups*

- Links to folders containing group-specific policies and company policies are inside each user's home directory.
- Temps can only write up to 30 MB worth of data onto each server, with a soft limit at 25 MB (exceeding this will lead to warnings.)

4.2 Script Information

Scripts can be downloaded from the following link, should it be necessary:
https://dev.hollylotor.online/cis245/scripts

4.2.1 *batchacct.py*

The script expects a file with one user's name per line. The script must be invoked at the command line, and must not be used as a module to another Python script. The script must be run as root, or with the usage of `sudo`.

Sample Input: `python3 batchacct.py FileWithNames.txt`
Sample Output:

```
Accounts Created
=====================================
s.username1000 | d89Bl24XolId24q0bQl97Yba | administrators
```

Remember: the `batchacct.py` script, for every six users in the input file, will assign the first to Admin, second and third to Developer, fourth and fifth to Staff, and the sixth to Temp.

4.3 Pre-Configuration

Nine folders were made in the `/etc/` folder: `skel-0`, `skel-2`, `skel-4`, `skel-5`, `policies-0`, `policies-2`, `policies-4`, `policies-5`, and `policies-all`. 0 is for administrators, 2 is for developers, 4 is for staff, 5 is for temps, and all is selfexplanatory. Within each `skel` folder, symbolic links to the corresponding policy folder as well as the `policies-all` folder were made.

For quotas, `quota` and `quotatool` were installed on server Lycanroc (the Ubuntu Server 18.04 server.) Nothing additional needed to be installed for server Buizel (the CentOS 8 server.) The fstab file was modified to add `usrquota` or `quota` to the appropriate disks (see C.1 for more info), and the command `sudo quotaon /dev/sda2` was used to enable quotas on Ubuntu (this wasn't necessary on CentOS.)

The password policy was created through the configuration changes specified in section C.1, before new users were added. Additionally, `libpam-pwquality` was installed on Ubuntu, as it was necessary to enforce the lower/uppercase and numeric requirements.

5 NETWORKING

(To Be Completed)

6 FIREWALL

(To Be Completed)

7 SECURITY

(To Be Completed)

7.1 Pubkey Authentication

Note: If you've already made a public/private key pair, you can save a copy of the public key into `~/.ssh/authorized_keys`*, then you should be able to log in with the key right away.*

Make sure you're running as the user you wish to log in as - this **shouldn't** be root, but rather an administrator account that can use `sudo` to run programs as root. As this administrator account, run the following command to create a private and public key with the comment "My Server Key":

```
$ ssh-keygen -t rsa -b 4096 -C "My Server Key"
```

Leave the save location as the default. Make sure you give it a password for added security. Now, with the keypair saved to `~/.ssh/id_rsa(.pub)`, copy the public key into a file called `authorized_keys`, which tells ssh that you should able to log in with that private key:

```
$ cp ~/.ssh/id_rsa.pub ~/.ssh/authorized_keys
```

and then use `scp` from your local PC to download the private key to your computer.

```
$ scp user@the.server.ip.here:/home/user/.ssh/id_rsa ~/.ssh/id_rsa
```

Make sure the new file has the correct permissions, and then you should be able to log in!

```
$ chmod 0600 ~/.ssh/id_rsa
$ ssh -i ~/.ssh/id_rsa user@the.server.ip.here
```

7.2 Restricting Logins

Once logged into the user with your private key via SSH, make sure that `sudo` still works in SSH.

```
$ sudo su -
```

Type in your user password when prompted. You should be running as root now, and the end of the prompt should have a pound symbol (#). At this point, you should be OK to restrict login access. With your favorite text editor, open up `/etc/ssh/sshd_config` and add the following lines to the bottom:

```
PermitRootLogin no
PasswordAuthentication no
```

Note: To save yourself a potential headache, make sure that any other lines that start with `PermitRootLogin` or `PasswordAuthentication` in the configuration file have a pound symbol (#) at the start to comment them out.

Save the file, then type the following command:

```
# systemctl restart sshd
```

Once done, exit completely out of the SSH session. Attempt to log into your user with a password and note that it no longer works over SSH. Then, attempt to log into your user with the private key you made earlier, and then use sudo to become root. If you're able to become root in your SSH session, congratulations, your server's now completely set up with pubkey authentication and you didn't lock yourself out of root. If not, uh... I hope you have local access.

7.3 fail2ban

All of the `fail2ban` configuration files live within the `/etc/fail2ban/` folder. There, the most interesting file you'll probably find is `jail.conf`. This has a large smattering of different "jails" you can set up, which are monitors that take certain actions when an IP breaks a particular rule. It's unwise to edit this file, though - future updates may break it. On the CentOS server, we'll need to make a new file in the `jail.d` folder - I chose `jail.d/jail.local`. On the Ubuntu server, we can modify the `/etc/fail2ban/jail.d/defaults-debian.conf` file already there.

I want to activate the SSH jail, and modify it to use the `ddos` mode. This enables monitoring the SSH port for failed logins, and uses certain rules to help protect against DDoS attacks. To do this, I'm going to add the following lines to our new file (or modify the existing file to match, in the case of Ubuntu):

```
[sshd]
mode = ddos
enabled = true
```

Afterwards, I restart fail2ban, and we're all good.

```
$ sudo systemctl restart fail2ban
```

8 CONTAINERS

(To Be Completed)

A COMMANDS AND SCRIPTS

B INSTALLED APPLICATIONS

Note: All dependencies for the below applications were also installed, but were not listed to avoid turning this document into a huge mess (only programs that were explicitly installed are mentioned.) Additionally, programs that were already included with the base installation (i.e. vim) were not included.

B.1 Both

- 6 MAR 2020 - openssh-server
- 6 MAR 2020 - tmux
- 6 MAR 2020 - emacs
- 6 MAR 2020 - fail2ban
- 6 MAR 2020 - cowsay
- 6 MAR 2020 - lolcat

B.2 Buizel (CentOS Linux 8)

- 6 MAR 2020 - epel-release
- 6 MAR 2020 - yum-utils
- 6 MAR 2020 - ruby
- 6 MAR 2020 - git
- 8 MAR 2020 - tcsh

B.3 Lycanroc (Ubuntu 18.04 LTS)

- 6 MAR 2020 - apt-transport-https
- 6 MAR 2020 - gnupg-agent
- 8 MAR 2020 - quota
- 8 MAR 2020 - quotatool
- 8 MAR 2020 - csh
- 8 MAR 2020 - libpam-pwquality

C CONFIGURATION

C.1 Modified/Added

C.1.1 *Both*

Note: The following was completed on 6 Mar 2020.

- Created account `holly`

- File `/home/holly/.ssh/authorized_keys` created
  Note: For security, the corresponding private key is not included in this document.

  ```
  ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQCmyKVl/hEji2DqYoXTxeqHCO5
  MqOadt7mNrTbl6VhCSNoV4J7eM4gIOOMLkkkIAPEgp0VgH3sVYEXxznDYd2PrGv
  nP/QOm+eq31mSQXfcDfkLlgUsIfehWqEwQqArtiNraEBo7tCuWMJEGLi+RXzJ++
  iJxER86/MJcu2N6ERixF/psZ9Nqu+u6/p3RLgWTf7chvIk9RdnYxseQrnUQtDXy
  g1FUBguyMFzCgB66qw5Cjxj3PDvXx35UOLdO3EWjuu0B1aDqRDlQenw1ZA1AFwt
  lztqzxXQhHc1InV+07VttS/K4iTqL454uFOiSB+bgl6+WzE8QB+Dv/KAV54DMzi
  1Gm/XIbs4rr7FE2XGehPxTwklvIbl1w22DYSM/47VHfsGMwjSivcerD16j5PIgJ
  j6XDwvdaC56F/L6tKGXt9BUiqvCI62wMjtpfB686YWKUrEBaV2zA4DPKn+XF9t9
  Oa5Omf6voLvTFwnrPB9VaPD7nGT/xZYs/Vp8SJx7rMv8XoIoF0AT5Yll4gfmoMl
  up7sDtW40KWQPH8GCu4YHu9JzgnxLT0ci/rM48+cHBOhtG/+MG59J+bxlM/Pcq+
  7Tdxi10myk9q7zXIAarTdFckTTdrkOhNdtpIz36f5EvAYSrf2WJPpCzFS0ltuPb
  3khTLs4P1tXIbz2yNfHMmQSfnqLXQ== holly@server.acme
  ```

- File `/etc/ssh/sshd_config` edited
  The following was appended to the end of the document, and any conflicting entries were removed:

  ```
  PermitRootLogin no
  PasswordAuthentication no
  ```

Note: The following was completed on 8 Mar 2020.

- Appended to `/etc/visudo`:

  ```
  %administrators ALL=(ALL) ALL
  ```

- Modified lines in `/etc/login.defs`:

  ```
  PASS_MAX_DAYS   180
  PASS_MIN_DAYS   0
  PASS_WARN_AGE   7
  ```

C.1.2 *Buizel (CentOS Linux 8)*

Note: The following was completed on 6 Mar 2020.

- Files/repositories added to `/etc/yum.repos.d/`:

  ```
  docker-ce.repo
  epel.repo
  epel-testing.repo
  epel-playground.repo
  epel-testing.repo
  ```

- File `/etc/fail2ban/jail.d/jail.local` added:

  ```
  [sshd]
  mode = ddos
  enabled = true
  ```

Note: The following was completed on 8 Mar 2020.

- Modified line in `/etc/fstab`:

  ```
  /dev/mapper/cl_server-home /home xfs defaults,quota 0 0
  ```

- Enabled quota in /home directory
- Modified password policy:

```
sudo authconfig --passminlen=12 --enablereqlower \
--enablerequpper --enablereqdigit --update
```

- Modified line in `/etc/pam.d/system-auth`:

```
password sufficient pam_unix.so sha512 shadow nullok
  try_first_pass use_authtok remember=100
```

C.1.3 *Lycanroc (Ubuntu Server 18.04 LTS)*

Note: The following was completed on 6 Mar 2020.

- Appended to `/etc/apt/sources.list`:

```
deb http://us.archive.ubuntu.com/ubuntu bionic-security \
multiverse
deb [arch=amd64] https://download.docker.com/linux/ubuntu \
$(lsb_release -cs) stable
```

- File `/etc/fail2ban/jail.d/defaults-debian.conf` modified:

```
[sshd]
mode = ddos
enabled = true
```

Note: The following was completed on 8 Mar 2020.

- Modified line in `/etc/fstab`:

```
/dev/disk/by-uuid/ababb95c-3fac-4667-9ac9-51c9be9c7347
  / ext4 defaults,usrquota 0 0
```

- Enabled quota on main disk
- Modified line in `/etc/pam.d/common-password`:

```
password [success=1 default=ignore] pam_unix.so obscure sha512
  minlen=12 ucredit=-1 dcredit=-1 ocredit=-1 remember=100
```

D ACCOUNT CREATION OUTPUT

Please refer to the attached document, ScriptOutput.txt.