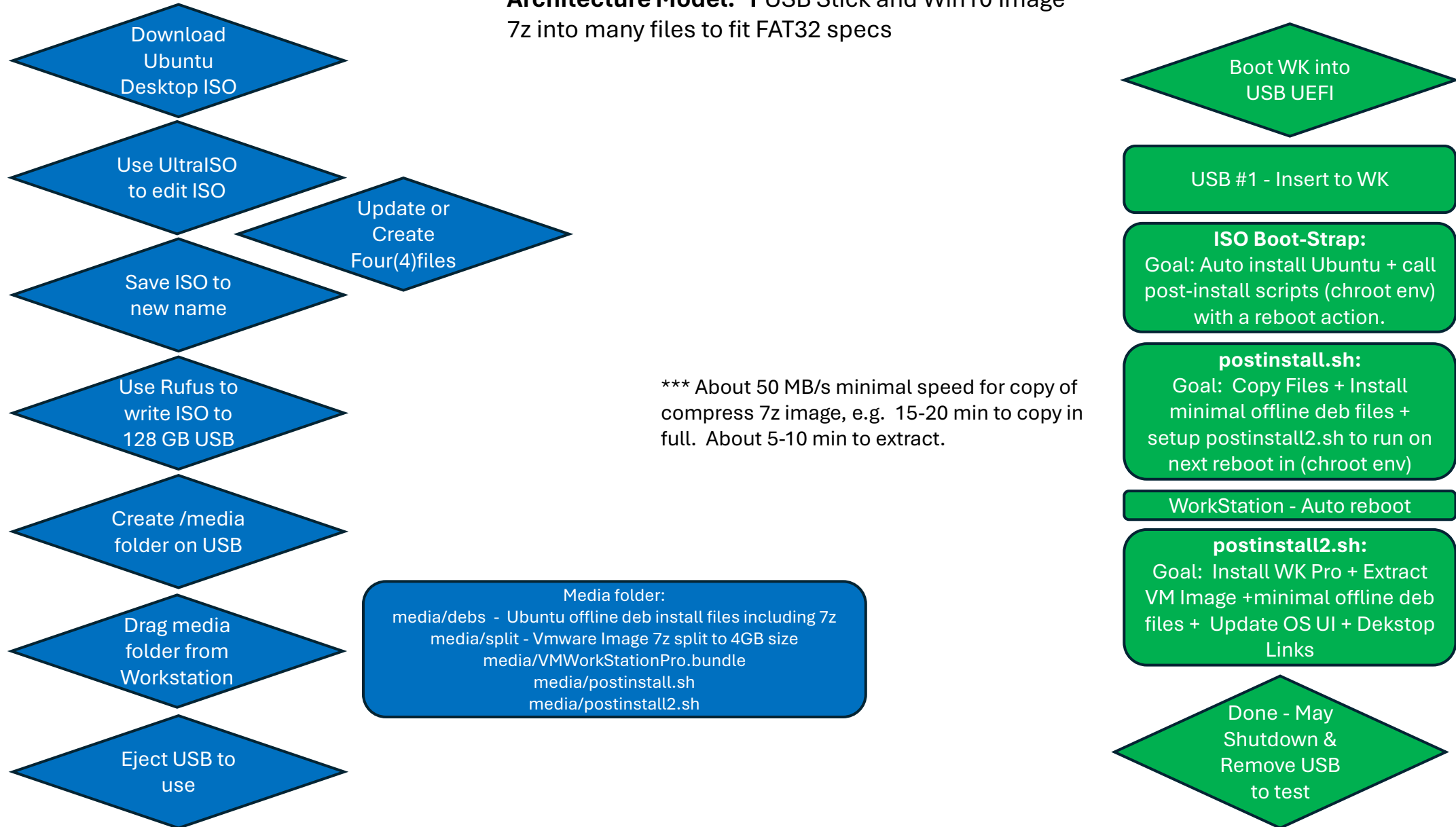


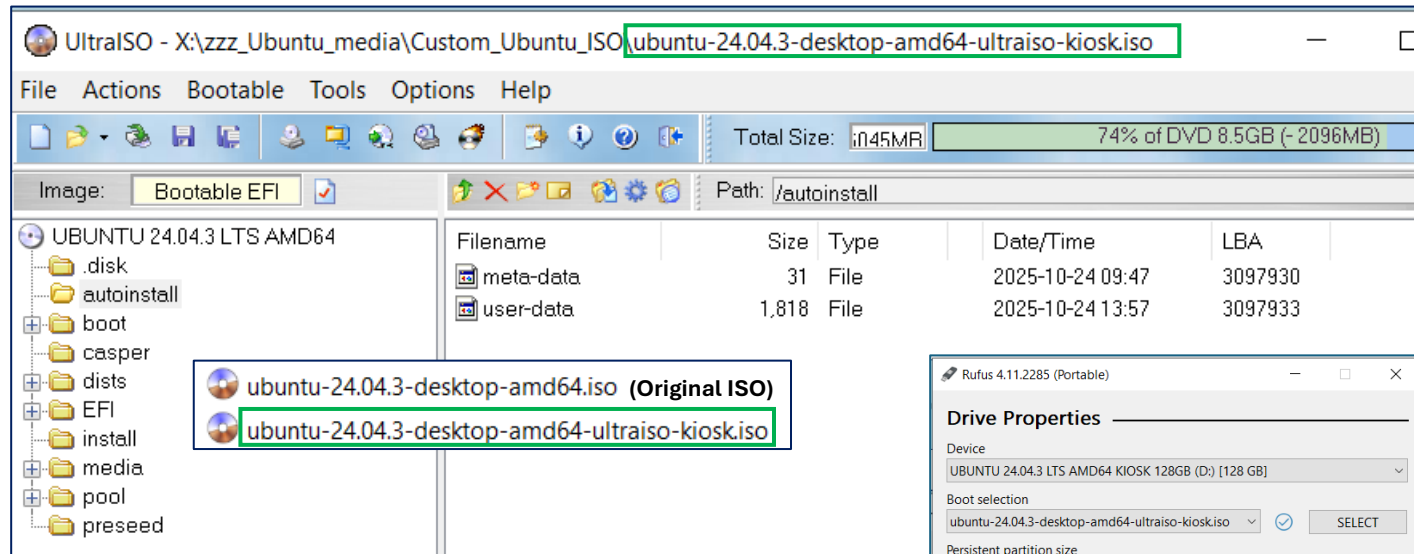
# Goal: Kiosk USB Stick with Ubuntu OS & Vmware Workstation Pro and MS Win X Image

- **Step 01: MS Win X image**
  - Create Win10 Image with software as-needed.
  - Enable autologin with 'netplwiz' MS Win control panel tool. Reboot & Confirm.
  - Remove any vmware snapshots or older memory files (.vmem) in the current install folder.
  - Defrag & Compress Disk Image using VMware disk settings to smallest size.
  - Use 7zip to compress file & name "win10.7z" & split to 4GB files to avoid FAT32 challenge on USB stick.
- **Step 02: Use UltraISO to edit Ubuntu ISO to create new bootable ISO with four (4) custom boot files.**
  - /autoinstall/user-data & /autoinstall/meta-data [create these files ahead of time]
  - /boot/grub/grub.cfg & /boot/grub/loopback.cfg [update these files ahead of time]
- **Step 03: Use Rufus with the new custom bootable ISO to a 64 GB USB stick.**
- **Step 04: Edit files on USB stick with Ubuntu**
  - Create a folder /media & a shell script under /media/postinstall.sh [This can be edited on the fly as needed to change installation.]
- **Step 05: Initial Validation of boot USB in a new system with auto-login (kiosk architecture)**
  - Ubuntu should auto-install with no internet access. [May test with no files under /media - to ensure it can auto-install the os]
  - Ubuntu desktop UI should auto-login.
  - No other files are available, but we are looking for basic confirmation first.
- **Step06: Copy the following install files to /media [These files will be installed by postinstall.sh & postinstall2.sh]**
  - 7z2501-linux-x64.tar.xz
  - VMware-Workstation-Full-25H2-24995812.x86\_64.bundle
  - win10.7z [the 10-20 4gb split files]
- **Step07: Validate the boot USB in a new system with auto-login (kiosk architecture) with post-install process.**
  - Ubuntu should auto-install with no internet access.
  - Ubuntu desktop UI should auto-login.
  - VMware Pro UI should load with the MS Win 10 image
  - MS Win 10 image should auto-start and login to desktop with no interaction.

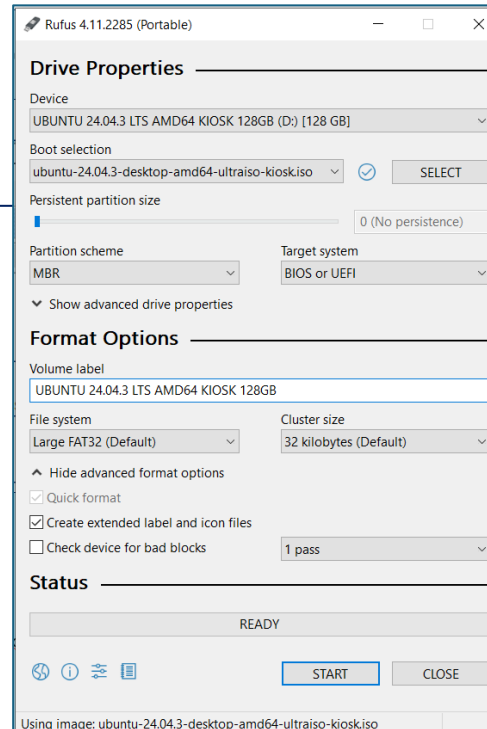
**Architecture Model: 1 USB Stick and Win10 Image**  
7z into many files to fit FAT32 specs



# Process & Tools Used to create Ubuntu Kiosk USB



1. **Ubuntu Desktop ISO** (Latest/Standard with UI)
2. **UltraISO** (ISO editor)
  - a. Edit the ISO's boot files
    - a. /boot/grub/grub.cfg
    - b. /boot/grub/loopback.cfg
    - c. /autoinstall/meta-data
    - d. /autoinstall/user-data [4 last-cmds]
3. **Rufus** (ISO->USB)
  - a. Write ISO to USB Flash Drive w/ UEFI
4. **postinstall.sh** (Bash Shell Script in /media)
  - a. Copy files from /media to SDD
  - b. Install offline packages: /media/debs/(.deb files)
  - c. Create SWAP file
  - d. Auto reboot
5. **postinstall2.sh** [systemd one-shot upon 1<sup>st</sup> reboot]
  - a. Install any other item we wish (Vmware Wk Pro)
  - b. Update as needed.



Name	Date modified	Type	Size
7zip_23.01+dfsg-11_amd64.deb	10/24/2025 11:36 ...	DEB File	1,803 KB
binutils_2.42-4ubuntu2.5_amd64.deb	10/24/2025 11:36 ...	DEB File	18 KB
binutils-common_2.42-4ubuntu2.5_amd64.deb	10/24/2025 11:36 ...	DEB File	235 KB
binutils-x86_64-linux-gnu_2.42-4ubuntu2.5_amd64.deb	10/24/2025 11:36 ...	DEB File	2,405 KB
build-essential_12.10ubuntu1_amd64.deb	10/24/2025 11:36 ...	DEB File	5 KB
bzip2_1.0.8-5.1build0.1_amd64.deb	10/24/2025 11:36 ...	DEB File	34 KB
dkms_3.0.11-1ubuntu13_all.deb	10/24/2025 11:36 ...	DEB File	51 KB
dpkg-dev_1.22.6ubuntu6.5_all.deb	10/24/2025 11:36 ...	DEB File	1,050 KB
fakeroot_1.33-1_amd64.deb	10/24/2025 11:36 ...	DEB File	66 KB
g++_4%3a13.2.0-7ubuntu1_amd64.deb	10/24/2025 11:36 ...	DEB File	2 KB
g++-13_13.3.0-6ubuntu2~24.04_amd64.deb	10/24/2025 11:36 ...	DEB File	16 KB
g++-13-x86-64-linux-gnu_13.3.0-6ubuntu2~24.04_amd64.deb	10/24/2025 11:36 ...	DEB File	11,877 KB
g++-x86-64-linux-gnu_4%3a13.2.0-7ubuntu1_amd64.deb	10/24/2025 11:36 ...	DEB File	1 KB
gcc_4%3a13.2.0-7ubuntu1_amd64.deb	10/24/2025 11:36 ...	DEB File	5 KB
gcc-13_13.3.0-6ubuntu2~24.04_amd64.deb	10/24/2025 11:36 ...	DEB File	483 KB
gcc-13-x86-64-linux-gnu_13.3.0-6ubuntu2~24.04_amd64.deb	10/24/2025 11:36 ...	DEB File	20,595 KB
gcc-x86-64-linux-gnu_4%3a13.2.0-7ubuntu1_amd64.deb	10/24/2025 11:36 ...	DEB File	2 KB
libalgorithm-diff-perl_1.201-1_all.deb	10/24/2025 11:36 ...	DEB File	41 KB
libalgorithm-diff-xs-perl_0.04-8build3_all.deb	10/24/2025 11:36 ...	DEB File	11 KB
libalgorithm-merge-perl_0.08-5_all.deb	10/24/2025 11:36 ...	DEB File	12 KB
libasan8_14.2.0-4ubuntu2~24.04_amd64.deb	10/24/2025 11:36 ...	DEB File	2,961 KB

Note: six (6) custom files for this architecture

- Four (4) embedded in the ISO image
- Two (2) external on USB under /media

```

└─ autoinstall
  └─ meta-data 1 instance-id: ubuntu-autoinstall
  └─ user-data
└─ boot
  └─ memtest86+x64.bin
  └─ grub
    └─ grub.cfg
    └─ loopback.cfg
└─ media
  └─ postinstall.sh
  └─ postinstall2.sh
  └─ vmware-workstation-pro-25h2.pdf
  └─ VMware-Workstation-Full-25H2-24995812.x86_64.bundle
└─ debs (offline packages)
  └─ 7zip_23.01+dfsg-11_amd64.deb
  └─ dkms_3.0.11-1ubuntu13_all.deb
  └─ build-essential_12.10ubuntu1_amd64.deb
  └─ linux-headers-6.8.0-86-generic_6.8.0-86.87_amd64.deb
  └─ linux-headers-6.8.0-86_6.8.0-86.87_all.deb
  └─ linux-headers-generic_6.8.0-86.87_amd64.deb
  └─ gcc-13_13.3.0-6ubuntu2~24.04_amd64.deb
  └─ g++-13_13.3.0-6ubuntu2~24.04_amd64.deb
  └─ make_4.3-4.1build2_amd64.deb
  └─ yamllint_1.33.0-1_all.deb
  └─ ... (binutils, libs, notes.txt, etc.)
└─ split (Windows VM archive parts)
  └─ win10.7z.001
  └─ win10.7z.002
  └─ win10.7z.003
  └─ win10.7z.004
  └─ win10.7z.005
  └─ win10.7z.006
  └─ win10.7z.007
  └─ win10.7z.008
  └─ win10.7z.009
  └─ win10.7z.010
  └─ win10.7z.011
  └─ win10.7z.012

```

```

1 #cloud-config
2 autoinstall: true
3 version: 1
4 identity: {}
5 ---hostname: ubuntu
6 ---username: ubuntu
7 ---# password: ubuntu (SHA-512 crypt)
8 ---#
9 ---# To generate a new SHA-512 password hash for this field:
10 ---# 1. Boot any Linux system (including Ubuntu Live USB)
11 ---# 2. Run this Command in a terminal:
12 ---#    mkpasswd --sha-512
13 ---# or, if 'mkpasswd' isn't installed:
14 ---#    python3 -c "import crypt; print(crypt.crypt(input('Password: '), crypt.mksalt(crypt.METHOD_SHA512)))"
15 ---# 3. Copy the resulting string (it starts with $6$...) into the 'password:' line below.
16 ---#
17 password: "$6$rounds=4096$ubuntus09NjptXD0pah3aHRjwM6e6r1kdFmsrQoDyETvmE9ztvtDtdVjNIEF5hSrGD4D2Jv0Z/JOfGbtOa2h4f4u4c1"
18
19 locale: en_US
20 timezone: America/Chicago
21 keyboard: {}
22 layout: us
23
24 ssh: {}
25 install-server: false
26
27 storage: {}
28 layout: {}
29 ---k name: direct
30 ---swap: {}
31 ---size: 0 ---# prevent curtin swap bug; we create swap in postinstall
32 ---overwrite: true
33
34 packages: [] ---# offline: do not apt install anything during install
35 ---user-data: {}
36 ---disable-root: false
37
38 late-commands:
39 ---# 1) Copy + chmod inside the target (never fail)
40 ---#    curtin in-target --target=/target -- bash -c 'cp -f /odrom/media/postinstall.sh /root/postinstall.sh && chmod +x /root/postinstall.sh' || true
41 ---# 2) Run Phase 1 work INSIDE target, but never fail curtin (log rc)
42 ---#    curtin in-target --target=/target -- bash -c 'set +e; /root/postinstall.sh > /var/log/postinstall.log 2>&1; echo "[INFO] postinstall.sh (in-target) rc=$?" >> /var/log/postinstall.log; exit 0'
43 ---# 3) Schedule reboot from the LIVE environment (outside chroot)
44 ---#    Release /target so Subiquity can unmount cleanly; detach reboot.
45 ---#    bash -c 'cd /; fuser -km /target || true; umount -R /target 2>/dev/null || umount -lR /target 2>/dev/null || true; (nohup /sbin/reboot -f >/dev/null 2>&1 <&- &) &; exit 0'
46 ---# 4) Failsafe: ensure GDM autologin even if postinstall.sh didn't run
47 ---#    curtin in-target --target=/target -- bash -c 'mkdir -p /etc/gdm3 && printf '[daemon]\nAutomaticLoginEnable=true\nAutomaticLogin=ubuntu\n' > /etc/gdm3/custom.conf || true'

```

```

1 set default=0.00
2 set timeout=10.00
3 set timeout_style=menu.00
4
5 loadfont unicode.00
6
7 set menu_color_normal=white/black.00
8 set menu_color_highlight=black/light-gray.00
9
10 # Optional: show a warning banner on the menu.00
11 insmod gfxterm.00
12 terminal_output gfxterm.00
13 echo.00
14 echo '*** WARNING: AUTOINSTALL WILL ERASE THE INTERNAL DISK ***'.00
15 echo 'Use "NO AUTOINSTALL" if you do NOT want to wipe the disk.'.00
16 echo 'Press e on an entry to edit and remove autoinstall if needed.'.00
17 echo.00
18 sleep 3.00
19
20 # Autoinstall (destructive).00
21 # Ensure the USB has /autoinstall/user-data and /autoinstall/meta-data.00
22 # Keep your helpers in /media/ (postinstall.sh, VMware bundle, 7-Zip tar, win10.7z).00
menuentry "Auto Install Ubuntu with VMware Workstation Pro in Kiosk Mode" {00
  ...echo "Autoinstall starting... internal disk will be overwritten.".00
  ...sleep 3.00
  ...set gfxpayload=keep.00
  ...linux ../casper/vmlinuz autoinstall 'ds=nocloud;s=cdrom/autoinstall' quiet text=---.00
  # ...linux ../casper/vmlinuz autoinstall 'ds=nocloud;s=cdrom/autoinstall' quiet splash=---.00
  ...initrd ../casper/initrd.00
}00
00

```

```
1 menuentry "Kiosk Ubuntu AutoInstall" {  
2     ...set gfxpayload=keep  
3     ...linux ../casper/vmlinuz autoinstall 'ds=nocloud;s=/cdrom/autoinstall' quiet text ---  
4     #...linux ../casper/vmlinuz autoinstall 'ds=nocloud;s=/cdrom/autoinstall' quiet splash ---  
5     ...initrd ../casper/initrd  
6 }  
7  
8 menuentry "Ubuntu (safe_graphics)" {  
9     ...set gfxpayload=keep  
10    ...linux ../casper/vmlinuz nomodeset autoinstall 'ds=nocloud;s=/cdrom/autoinstall' quiet splash text ---  
11    ...initrd ../casper/initrd  
12 }
```

Goals: Install the OS then run a primary post-install script (that runs in chroot env) & reboots.

- UEFI will load grub.cfg / loopback.cfg first.
- These files will load the /autoinstaller folder.
- The "meta-data" and "user-data" will be referenced.
- The user-data's "late-commands" will then call "postinstall.sh" script.
  - The "user-data" file must be a YAML / UTF-8 / UNIX LF Return format.
- postinstall.sh (chroot env) will set a few configurations, then set postinstall2.sh to run on next reboot.
- postinstall2.sh will install software and desktop links

# Boot USB via UEFI



Boot mode is set to: UEFI; Secure Boot: OFF

LEGACY BOOT:

M.2 PCIe SSD  
Internal HDD  
USB Storage Device

UEFI BOOT:

UEFI: USB DISK 3.0 PMAP, Partition 1

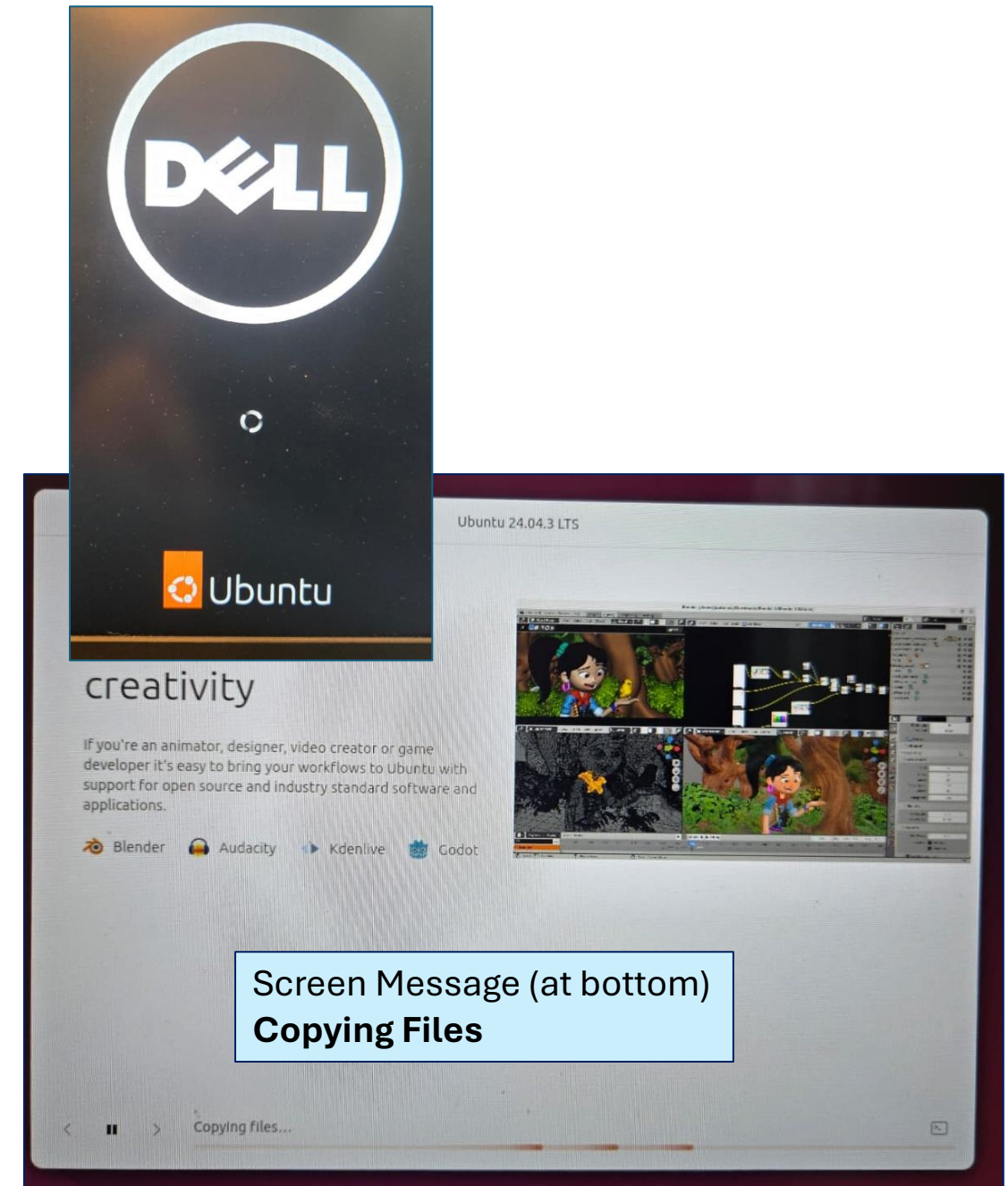
OTHER OPTIONS:

BIOS Setup  
BIOS Flash Update  
Diagnostics  
Change Boot Mode Settings

Autoinstall starting... Internal disk will be overwritten.

**Timing:** Dell Precision 7510 Laptop

- 1<sup>st</sup> boot: 12-15 min from UEFI Boot, to the installation of Ubuntu OS & install of offline deb files & copy of /media folder & auto-reboot.
- 2<sup>nd</sup> reboot: 15-20 min, install VMware Workstation pro & extract VMware image, configurations & request for reboot.
- 3<sup>rd</sup> reboot: 1-3 min, auto load VMware Workstation image.



# Prepare the MS Windows Vmware Image

## Command Prompt

Microsoft Windows [Version 10.0.19045.5965]  
(c) Microsoft Corporation. All rights reserved.

C:\Users\Administrator>netplwiz

C:\Users\Administrator>

### Auto login for MS Win 10

- Use either netplwiz or
- SysInternal Tools: Autologon64.exe

## User Accounts

Users Advanced



Use the list below to grant or deny users access to your computer, and to change passwords and other settings.

☐ Users must enter a user name and password to use this computer.

Users for this computer:

User Name	Group
Administrator	Debugger Users; docker-users; ...

Add...

Remove

Properties

Password for Administrator



To change your password, press Ctrl-Alt-Del and select Change Password.

Reset Password...

OK

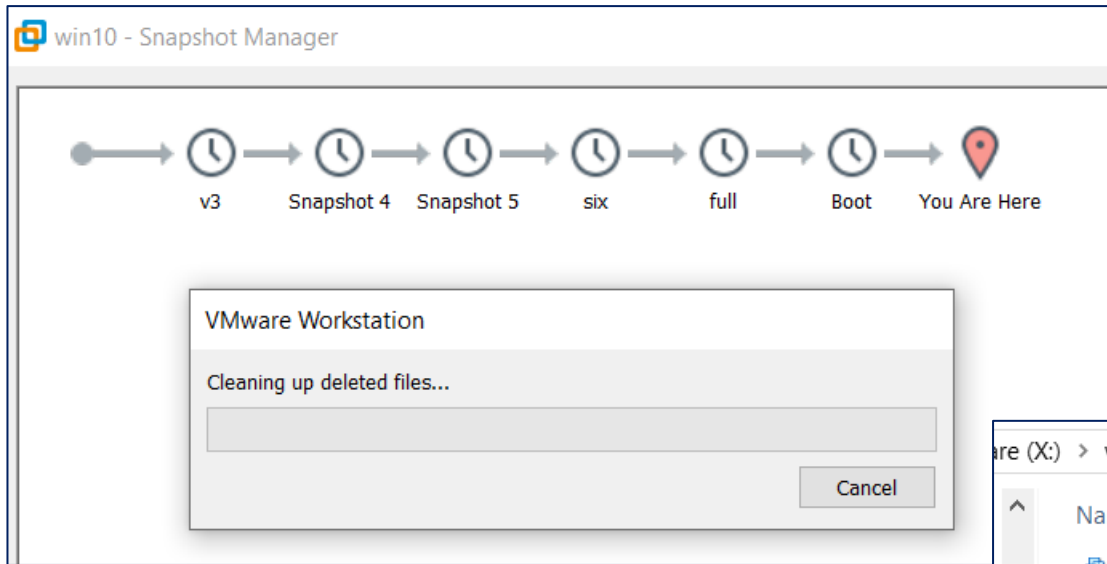
Cancel

Apply



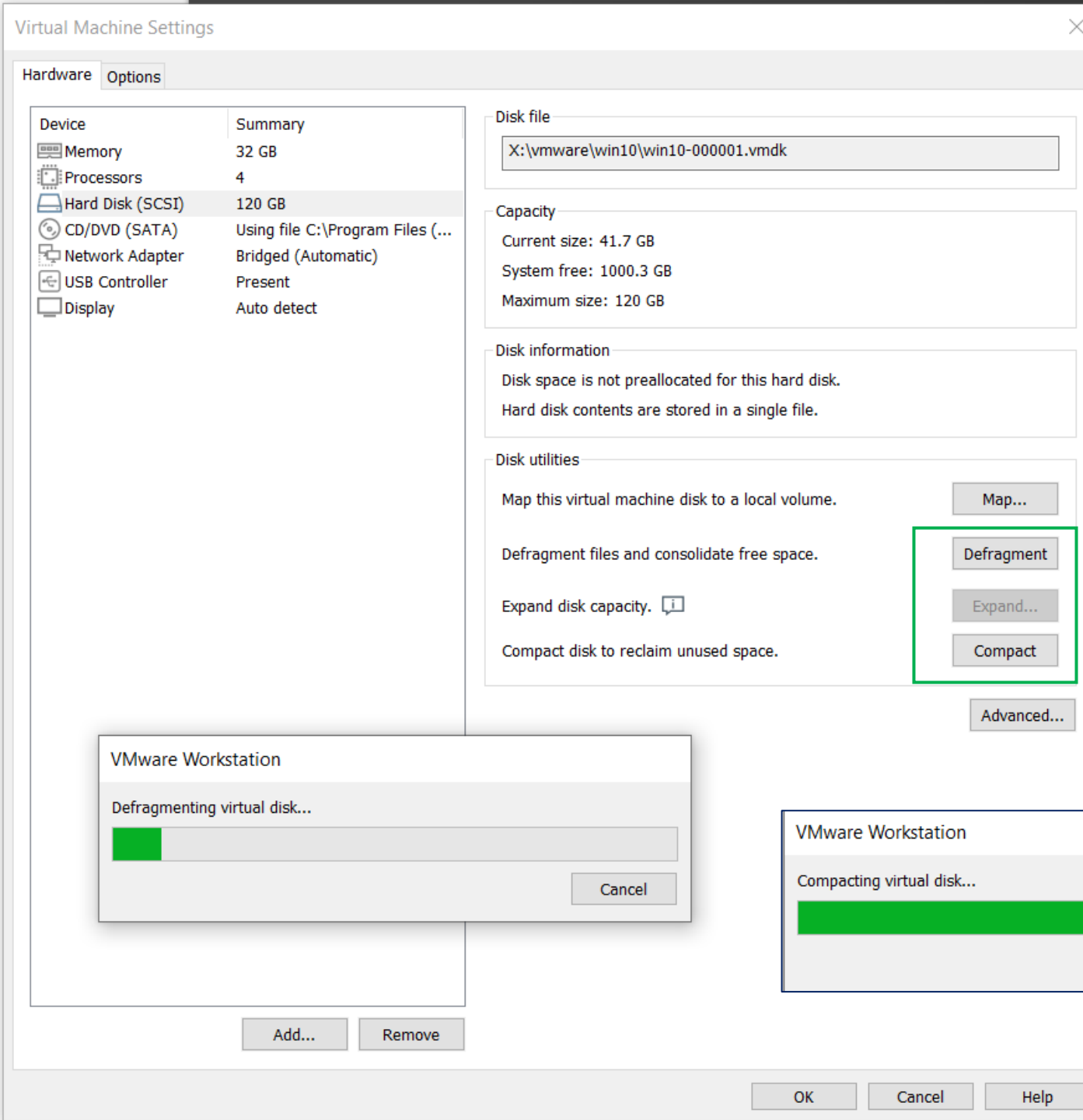
To reduce size:

- If there are any VMware SNAPSHOTS, please delete them prior to defrag/compression efforts.
  - Optionally: Clone current version to a new deployment with no snapshots.
- Delete any "old" suspended memory files (.vmem) (back up prior to any deletion)



The screenshot shows a Windows File Explorer window displaying the contents of the 'vmware > win10' directory. The file 'win10-7c681510.vmem' is selected, and a tooltip shows its details: Type: VMEM File, Size: 32.0 GB, Date modified: 9/26/2024 12:01 PM.

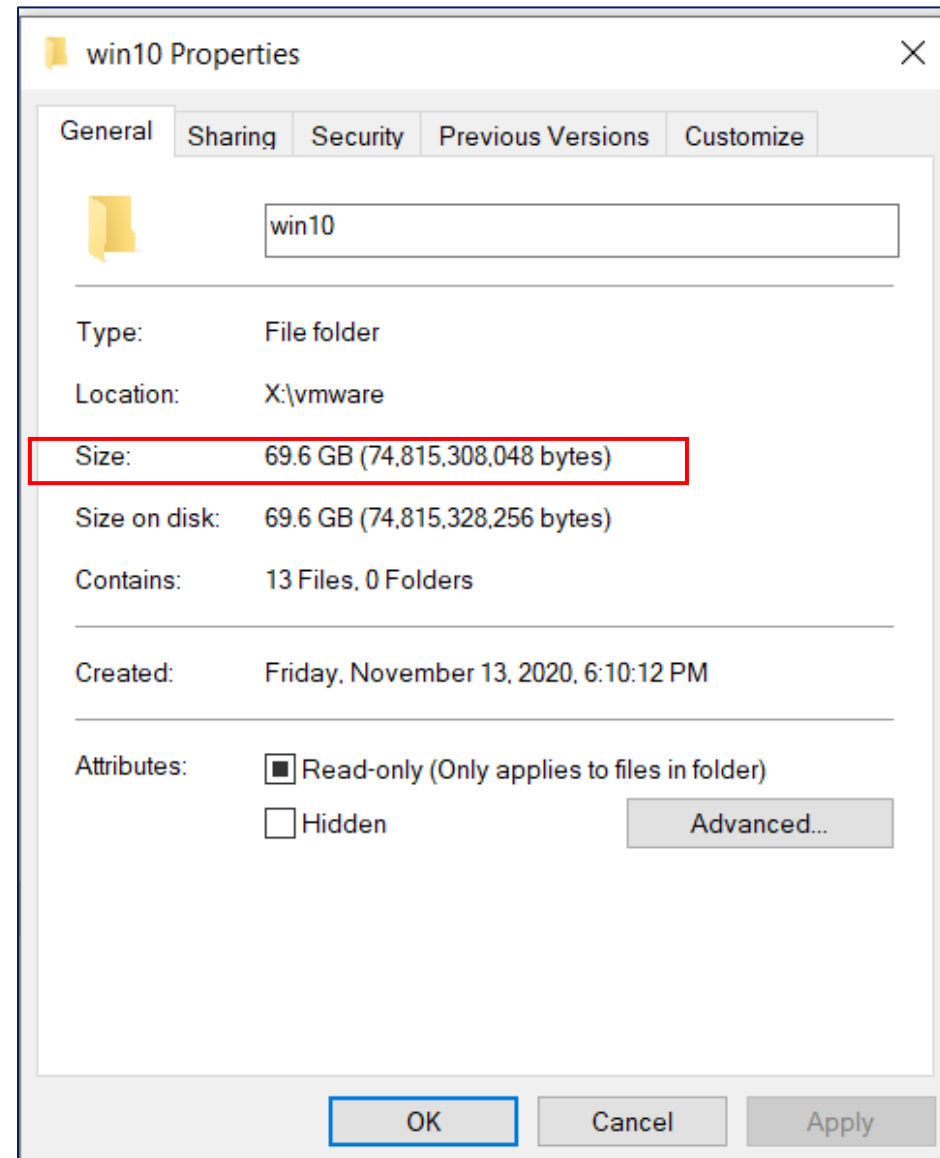
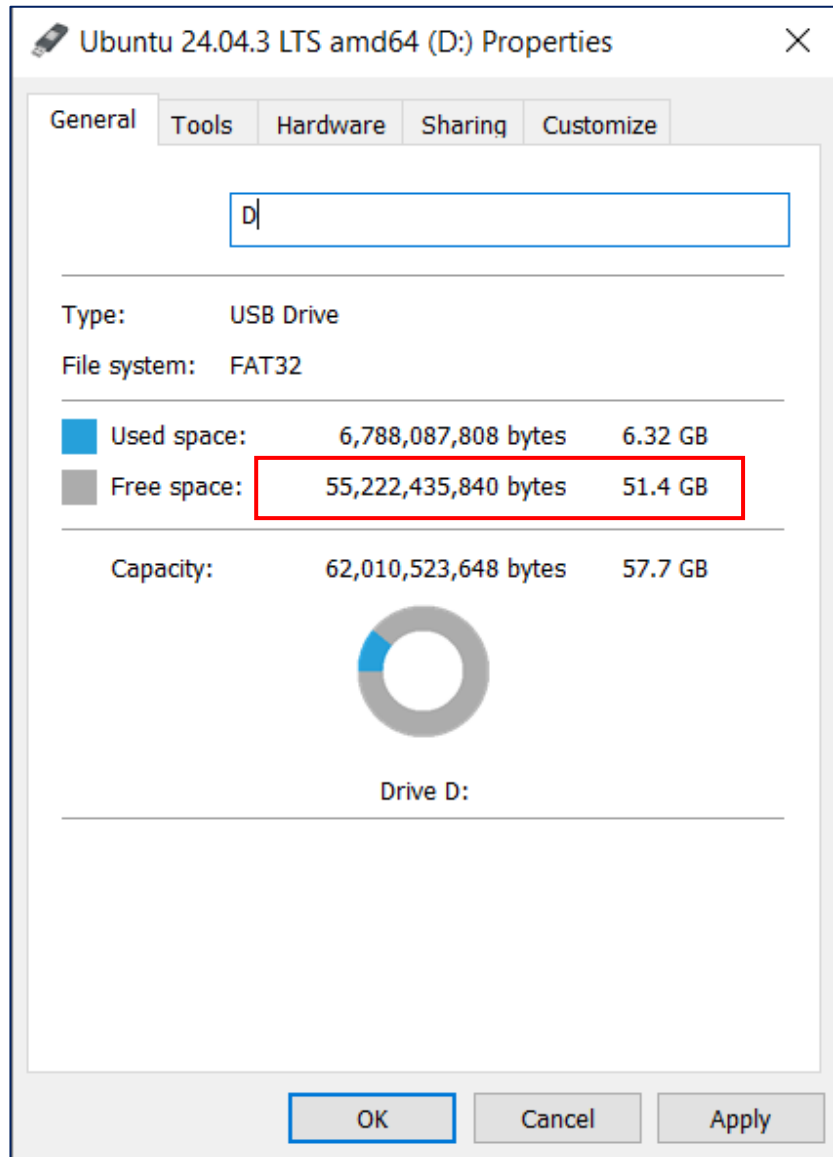
Name	Date modified	Type	Size
win10.vmdk	10/23/2025 3:29 PM	VMware virtual dis...	71,233,600 KB
win10-7c681510.vmem	9/26/2024 12:01 PM	VMEM File	33,554,432 KB
win10-000001.vmdk		re virtual dis...	23,799,872 KB
win10-000007.vmdk		re virtual dis...	15,424 KB
vmware-vmx.dmp		file	12,147 KB
vmmcores-1.gz	9/26/2024 12:02 PM	GZ File	10,605 KB
win10-7c681510.vms	9/26/2024 12:02 PM	VMware suspende...	8,834 KB
vmware-1.log	6/30/2025 2:43 PM	Text Document	3,094 KB
vmware-2.log	6/13/2025 5:37 PM	Text Document	1,010 KB



Defrag and Compress the Hard Drive for  
Vmware Image to fit on USB stick

- We are aiming for a compressed image  
folder less than 50 GB

Ubuntu UBS 64 GB Boot has 51 GB free space.  
Current Vmware Image Folder size is greater.



# 7zip notes

Use 7zip & adjust settings -

- Select "**Non-Solid**" for speed on extraction
- Spilt to **4 GB** Files (to avoid FAT32 challenge)

### Add to Archive

Archive: X:\zzz\_Ubuntu\_media\media\  
win10\_2.7z

Archive format: 7z

Compression level: Normal

Compression method: LZMA2

Dictionary size: 32 MB

Word size: 32

Solid Block size: Non-solid

Number of CPU threads: 12 / 12

Memory usage for Compressing: 3888 MB

Memory usage for Decompressing: 34 MB

Split to volumes, bytes:

4092M - FAT

Parameters:

7-Zip

CRC SHA

010 Editor

Compare with Altova DiffDog

Share

WinMerge

Open archive

Open archive

Extract files...

Extract Here

Extract to "win10\"

Test archive

Add to archive...

Elapsed time: 00:06:22 Total size: 71349 M  
Remaining time: 00:39:21 Speed: 26 MB/s  
Files: 9 / 13 Processed: 9936 M  
Compression ratio: 52% Compressed size: 5243 M

Adding  
win10\  
win10.vmdk

Name	Date modified	Type	Size
win10.7z.001	10/25/2025 1:59 AM	001 File	4,190,208 KB
win10.7z.002	10/25/2025 1:18 AM	002 File	4,190,208 KB
win10.7z.003	10/25/2025 1:22 AM	003 File	4,190,208 KB
win10.7z.004	10/25/2025 1:26 AM	004 File	4,190,208 KB
win10.7z.005	10/25/2025 1:30 AM	005 File	4,190,208 KB
win10.7z.006	10/25/2025 1:34 AM	006 File	4,190,208 KB
win10.7z.007	10/25/2025 1:39 AM	007 File	4,190,208 KB
win10.7z.008	10/25/2025 1:44 AM	008 File	4,190,208 KB
win10.7z.009	10/25/2025 1:47 AM	009 File	4,190,208 KB
win10.7z.010	10/25/2025 1:52 AM	010 File	4,190,208 KB
win10.7z.011	10/25/2025 1:58 AM	011 File	4,190,208 KB
win10.7z.012	10/25/2025 1:59 AM	012 File	990,291 KB

Disk 3

Removable

57.77 GB

Online


UBUNTU 24.0 (D:)

57.77 GB FAT32

Healthy (Active, Primary Partition)

FAT32 does **NOT** allow files > 4GB

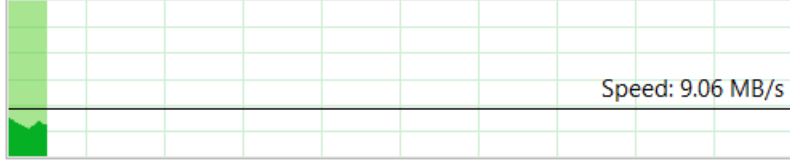
Final media folder state & test of media copy process speed.  
- Use USB specs 3.1/3.2 for performance for the USB stick.

Name	Date modified	Type	Size
7z2501-linux-x64.tar.xz	10/23/2025 1:31 PM	XZ File	1,535 KB
 gueststore-vmtools-13.0.5-0.24916190.tar.gz	10/23/2025 1:29 PM	GZ File	100,057 KB
VMware-Workstation-Full-25H2-24995812.x86_64.bundle	10/23/2025 1:28 PM	BUNDLE File	302,273 KB
win10_2.7z.001	10/23/2025 5:16 PM	001 File	4,190,208 KB
win10_2.7z.002	10/23/2025 4:48 PM	002 File	4,190,208 KB
win10_2.7z.003	10/23/2025 4:51 PM	003 File	4,190,208 KB
win10_2.7z.004	10/23/2025 4:54 PM	004 File	4,190,208 KB
win10_2.7z.005	10/23/2025 4:57 PM	005 File	4,190,208 KB
win10_2.7z.006	10/23/2025 5:00 PM	006 File	4,190,208 KB
win10_2.7z.007	10/23/2025 5:03 PM	007 File	4,190,208 KB
win10_2.7z.008	10/23/2025 5:07 PM	008 File	4,190,208 KB
win10_2.7z.009	10/23/2025 5:10 PM	009 File	4,190,208 KB
win10_2.7z.010	10/23/2025 5:13 PM	010 File	4,190,208 KB
win10_2.7z.011	10/23/2025 5:16 PM	011 File	4,190,208 KB
win10_2.7z.012	10/23/2025 5:16 PM	012 File	935,489 KB

4% complete

Copying 12 items from media to media

4% complete



Speed: 9.06 MB/s


Name: win10\_2.7z.001  
Time remaining: About 1 hour and 35 minutes  
Items remaining: 11 (42.6 GB)

⬆ Fewer details

0% complete

Copying 13 items from media to media

0% complete



Speed: 6.40 MB/s

Name: win10\_2.7z.003  
Time remaining: About 4 hours and 15 minutes  
Items remaining: 13 (44.4 GB)

⬆ Fewer details



# Rufus notes

Rufus 4.11.2285 (Portable)

### Drive Properties

Device  
Ubuntu 24.04.3 LTS amd64 (D:) [64 GB]

Boot selection  
ubuntu-24.04.3-desktop-amd64.iso

Persistent partition size  
0 (No persistence)

Partition scheme  
MBR

Target system  
BIOS or UEFI

Hide advanced drive properties

☐ List USB Hard Drives

☐ Add fixes for old BIOSes (extra partition, align, etc.)

☐ Enable runtime UEFI media validation

### Format Options

Volume label  
Ubuntu 24.04.3 LTS amd64

File system  
Large FAT32 (Default)

Cluster size  
32 kilobytes (Default)

Hide advanced format options

☒ Quick format

☒ Create extended label and icon files

☐ Check device for bad blocks

1 pass

### Status

READY

START CLOSE

Using image: ubuntu-24.04.3-desktop-amd64.iso

Use Rufus executable to create bootable USB stick.

- Rufus allows NTFS file system type, but it is not 100% for all newer BIOS UEFI to boot correctly, so we selected standard FAT32 file system.
- Avoid updating the DBX files for "secure boot" - not required for the kiosk architecture.

#### DBX update available



Rufus has found an updated version of the DBX files used to perform UEFI Secure Boot revocation checks. Do you want to download this update?

- Select 'Yes' to connect to the Internet and download this content
- Select 'No' to cancel the operation

Note: The files will be downloaded in the application's directory and will be reused automatically if present.

Yes

No

Rufus 4.11.2285 (Portable)

Drive Properties

Device

UBUNTU 24\_0 (D:) [64 GB]

Boot selection

ubuntu-24.04.3-desktop-amd64-ultraiso-kiosk.iso

SELECT

Persistent partition size

0 (No persistence)

Partition scheme

MBR

Target system

BIOS or UEFI

Show advanced drive properties

Format Options

Volume label

UBUNTU 24.04.3 LTS AMD64

File system

Large FAT32 (Default)

Cluster size

32 kilobytes (Default)

Hide advanced format options

Quick format

Create extended label and icon files

Check device for bad blocks

1 pass

Status

Copying ISO files: 30.1%

START

CANCEL

D:\casper\minimal.enhanced-secureboot.es.squashfs (2 MB)00:04:20

Rufus 4.11.2285 (Portable)

Drive Properties

Device

UBUNTU 24\_0 (D:) [64 GB]

Boot selection

ubuntu-24.04.3-desktop-amd64-ultraiso-kiosk.iso

SELECT

Persistent partition size

0 (No persistence)

Partition scheme

MBR

Target system

BIOS or UEFI

Show advanced drive properties

Format Options

Volume label

UBUNTU 24.04.3 LTS AMD64

File system

Large FAT32 (Default)

Cluster size

32 kilobytes (Default)

Hide advanced format options

Quick format

Create extended label and icon files

Check device for bad blocks

1 pass

Status

Copying ISO files: 80.4%

START

CANCEL

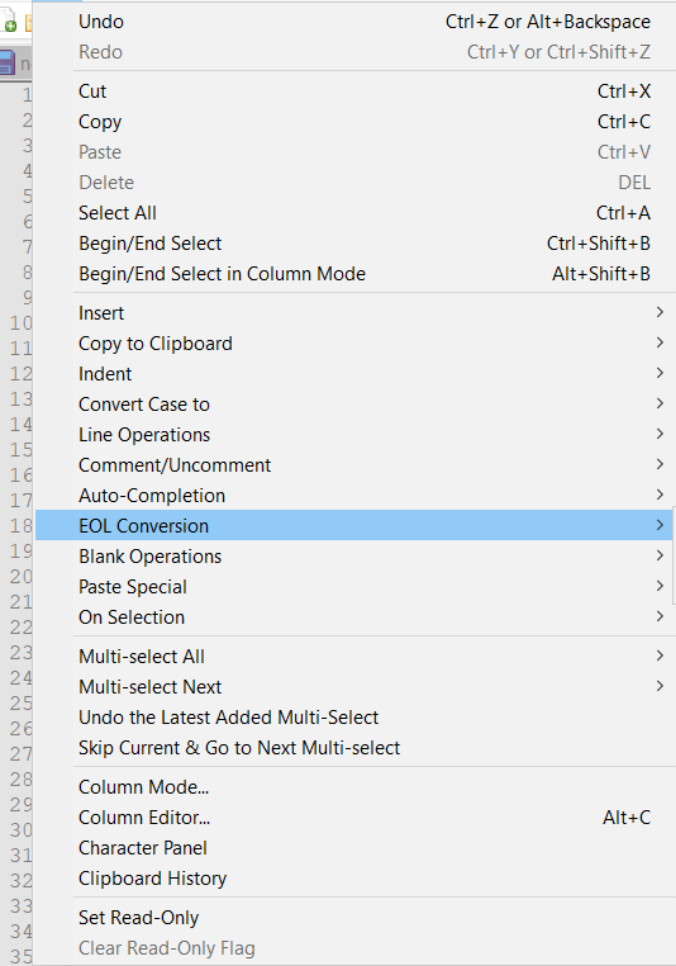
...\linux-modules-extra-6.14.0-27-generic\_6.14.0-27.27~24.04.1\_amd64.d...00:11:37

# Format Notes

Ensure all files are UNIX LF (line feed) ending characters.

- Use View / Show Symbol / Select Space & Tab & Show End of Line
- Use Edit / EOL Conversion / Unix (LF) to convert.
- Validate the Encoding is set to: UTF-8 Encoding w/o BOM

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?



bt\_vendor.conf new 6

```
for this field:
ntu Live USB)
```

```
crypt.crypt(input('Password: '), crypt.mksalt(crypt.METHOD_SHA512))) "
with $6$... into the 'password:' line below.
```

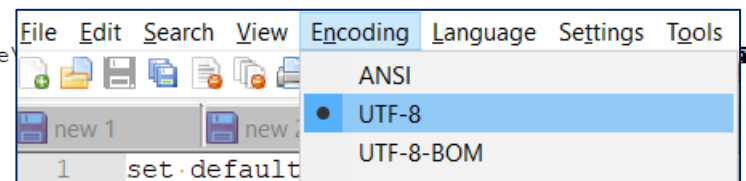
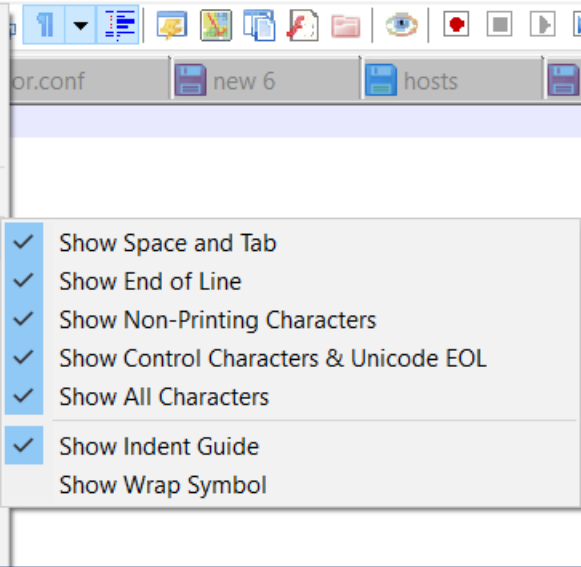
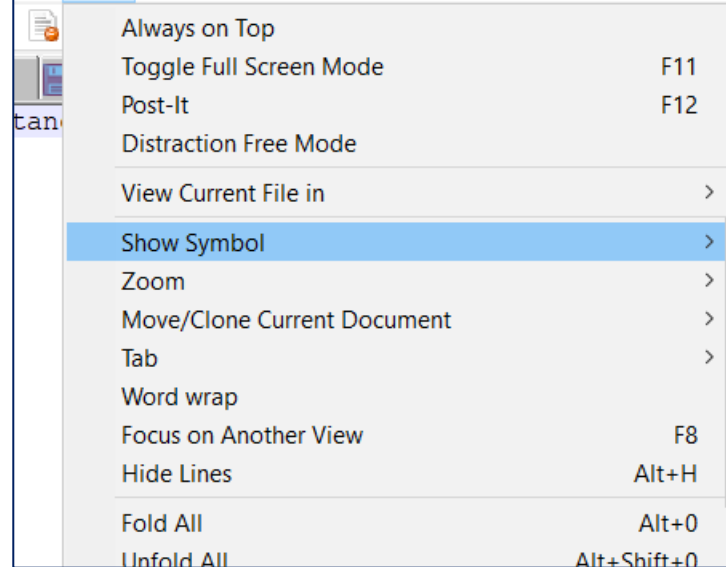
```
Onah3aHRIwM6e6r1kdDPmsrQoDyETvmE9ztvjDtdVjNIEF5hSrGD4D2Jv0Z/JOFGbtOa2h4f4u4c1"
```

```
linux-headers-generic
xz-utils
p7zip-full
```

```
user-data:
disable_root: false
late-commands:
```

```
curtin in-target --target=/target.chmod +x /cdrom/media/postinstall.sh
curtin in-target --target=/target.bash /cdrom/media/postinstall.sh > /target/var/log/postinstall.log 2>&1
# Failsafe: ensure GDM autologin even if postinstall.sh doesn't run
curtin in-target --target=/target.bash -c "mkdir -p /target/etc/gdm3. && echo -e '[daemon]\nAutomaticLoginEnable=true'
```

Search View Encoding Language Settings Tools Macro Run Plugins Window ?



postinstall.sh & postinstall2.sh



# postinstall.sh (runs in a chroot env - so we can only do selected updates)

```
1 #!/usr/bin/env bash
2 #
3 # kiosk postinstall.sh - Phase 1 (always executed inside curtin chroot)
4 # -- Stages offline media to /opt/usb-seed
5 # -- Installs any .deb payloads (best-effort)
6 # -- Creates swapfile
7 # -- Enables GDM autologin + disables lock screen
8 # -- Registers Phase 2 systemd oneshot
9 # -- Always exits 0 so Subiquity never fails (reboot handled by user-data)
10 #
11 VERSION=1.3.3
12 BUILD_DATE=2025-10-29
13
14 # relaxed error handling inside curtin chroot
15 set -e +o pipefail
16 umask 022
17 set -m
18 trap - ERR # disable ERR trap completely
19
20 LOG=/var/log/kiosk_postinstall_phase1.log
21 exec >> (tee -a "$LOG") 2>&1
22 _ts() { date +%F-%T; }
23 msg() { printf "[%s] %s\n" "${_ts}" "$*" ; }
24 warn() { printf "[%s] [WARN] %s\n" "${_ts}" "$*" ; }
25
26 msg "===== kiosk postinstall.sh start: (v${VERSION}) ${BUILD_DATE}) ====="
27
28 # basic config
29 #
30 USB_MEDIA=${USB_MEDIA:-/cdrom/media}
31 SEED_DST=${SEED_DST:-/opt/usb-seed}
32 DEB_DIR=${DEB_DIR:-${SEED_DST}/debs}
33 USER_NAME=${USER_NAME:-ubuntu}
34 SWAP_GB=${SWAP_GB:-8}
35
36 for v in USB_MEDIA SEED_DST DEB_DIR USER_NAME SWAP_GB; do
37     eval "msg '\${v}=${!v}'"
38 done
39
40 # stage offline media
41 #
42 msg "Staging offline media from $USB_MEDIA -> $SEED_DST"
43 install -d -m 0755 "$SEED_DST"
44 rsync -avh --delete --info=NAME,STATS "$USB_MEDIA" "$SEED_DST/" 2>&1
45 if [ $? -ne 0 ]; then
46     warn "rsync failed, trying cp -a fallback"
47     cp -a "$USB_MEDIA/" "$SEED_DST/" || warn "copy fallback failed"
48 fi
49 ls -la "$SEED_DST" || true
50
51 # install offline .debs (best effort)
52 #
53 if compgen -G "${DEB_DIR}//*.deb" >/dev/null 2>&1; then
54     msg "Installing .debs from $DEB_DIR"
55     apt-get update || warn "apt-get update failed (expected offline)"
56
```

```
58     apt-get update || warn "apt-get update failed (expected offline)"
59     dpkg -i "${DEB_DIR}//*.deb" || warn "dpkg -i non-zero"
60     apt-get -f install -y || warn "apt-get -f install non-zero"
61 else
62     msg "No .deb payloads under $DEB_DIR"
63 fi
64
65 # create swapfile
66 #
67 msg "Creating ${SWAP_GB} GB swapfile"
68 SWAPFILE=/swapfile
69 if ! grep -q 'swapfile' /etc/fstab 2>/dev/null; then
70     fallocate -l "${SWAP_GB}G" "$SWAPFILE" 2>/dev/null || {
71         dd if=/dev/zero of="$SWAPFILE" bs=1M count=$((SWAP_GB*1024)) status=progress
72         chmod 600 "$SWAPFILE"
73         mkswap "$SWAPFILE"
74         swapon "$SWAPFILE"
75         echo '/swapfile none swap sw 0 0' >> /etc/fstab
76     }
77     msg "Swapfile already present"
78 fi
79 swapon --show || true
80
81 # GDM autologin
82 #
83 msg "Configuring GDM autologin for ${USER_NAME}"
84 install -d -m 0755 /etc/gdm3
85 cat >/etc/gdm3/custom.conf <<EOF
86 [daemon]
87 AutomaticLoginEnable=true
88 AutomaticLogin=${USER_NAME}
89 EOF
90
91 #
92 # disable lock screen / idle
93 #
94 msg "Writing dconf defaults"
95 install -d -m 0755 /etc/dconf/db/local.d
96 cat >/etc/dconf/db/local.d/00-kiosk <<DCONF
97 [org/gnome/desktop/screensaver]
98 lock-enabled=false
99 [org/gnome/desktop/session]
100 idle-delay=uint32 0
101 [org/gnome/settings-daemon/plugins/power]
102 sleep-inactive-ac-type='nothing'
103 sleep-inactive-battery-type='nothing'
104 power-button-action='nothing'
105 [org/gnome/desktop/lockdown]
106 disable-lock-screen=true
107 DCONF
108 dconf update || warn "dconf update non-zero"
109
110 #
111 # register phase 2 systemd oneshot
112 #
113 msg "Registering kiosk-postinstall2.service"
114 UNIT=/etc/systemd/system/kiosk-postinstall2.service
115 WRAP=/usr/local/sbin/kiosk-phase2.sh
116 install -d -m 0755 /usr/local/sbin /var/lib/kiosk
117 cat >${WRAP} <<WRAP
118 #!/usr/bin/env bash
119 set -e
120 LOG=/var/log/kiosk_postinstall_phase2.log
121 exec >> (tee -a "$LOG") 2>&1
122 echo "$(date +%F-%T) phase2 start"
123 [-f /opt/usb-seed/postinstall2.sh] && bash /opt/usb-seed/postinstall2.sh
124 date > /var/lib/kiosk/postinstall2.done
125 systemctl disable --now kiosk-postinstall2.service || true
126 echo "$(date +%F-%T) phase2 done"
127 WRAP
128 chmod 0755 "$WRAP"
129
130 cat >"$UNIT" <<UNIT
131 [Unit]
132 Description=Kiosk Phase 2 (VMware + VM extraction + tweaks)
133 After=network-online.target systemd-udev-settle.service
134 Wants=network-online.target
135 ConditionPathExists=!/var/lib/kiosk/postinstall2.done
136 [Service]
137 Type=oneshot
138 ExecStart=/usr/local/sbin/kiosk-phase2.sh
139 StandardOutput=journal+console
140 StandardError=journal+console
141 TimeoutStartSec=0
142 [Install]
143 WantedBy=multi-user.target
144 UNIT
145
146 systemctl --no-reload enable kiosk-postinstall2.service || warn "systemctl enable non-zero"
147
148 # completion stamp
149 #
150 mkdir -p /var/lib/kiosk
151 date -Isconds >/var/lib/kiosk/postinstall1.done
152 msg "Phase 1 complete; reboot will be invoked by user-data."
153 sync
154 udevadm settle --timeout=30 || true
155 msg "===== kiosk postinstall.sh end (exit 0) ====="
156 exit 0
157
```

```
102 [org/gnome/settings-daemon/plugins/power]
103 sleep-inactive-ac-type='nothing'
104 sleep-inactive-battery-type='nothing'
105 power-button-action='nothing'
106 [org/gnome/desktop/lockdown]
107 disable-lock-screen=true
108 DCONF
109 dconf update || warn "dconf update non-zero"
110
111 #
112 # register phase 2 systemd oneshot
113 #
114 msg "Registering kiosk-postinstall2.service"
115 UNIT=/etc/systemd/system/kiosk-postinstall2.service
116 WRAP=/usr/local/sbin/kiosk-phase2.sh
117 install -d -m 0755 /usr/local/sbin /var/lib/kiosk
118 cat >${WRAP} <<WRAP
119 #!/usr/bin/env bash
120 set -e
121 LOG=/var/log/kiosk_postinstall_phase2.log
122 exec >> (tee -a "$LOG") 2>&1
123 echo "$(date +%F-%T) phase2 start"
124 [-f /opt/usb-seed/postinstall2.sh] && bash /opt/usb-seed/postinstall2.sh
125 date > /var/lib/kiosk/postinstall2.done
126 systemctl disable --now kiosk-postinstall2.service || true
127 echo "$(date +%F-%T) phase2 done"
128 WRAP
129 chmod 0755 "$WRAP"
130
131 cat >"$UNIT" <<UNIT
132 [Unit]
133 Description=Kiosk Phase 2 (VMware + VM extraction + tweaks)
134 After=network-online.target systemd-udev-settle.service
135 Wants=network-online.target
136 ConditionPathExists=!/var/lib/kiosk/postinstall2.done
137 [Service]
138 Type=oneshot
139 ExecStart=/usr/local/sbin/kiosk-phase2.sh
140 StandardOutput=journal+console
141 StandardError=journal+console
142 TimeoutStartSec=0
143 [Install]
144 WantedBy=multi-user.target
145 UNIT
146
147 systemctl --no-reload enable kiosk-postinstall2.service || warn "systemctl enable non-zero"
148
149 # completion stamp
150 #
151 mkdir -p /var/lib/kiosk
152 date -Isconds >/var/lib/kiosk/postinstall1.done
153 msg "Phase 1 complete; reboot will be invoked by user-data."
154 sync
155 udevadm settle --timeout=30 || true
156 msg "===== kiosk postinstall.sh end (exit 0) ====="
157 exit 0
158
```

# postinstall2.sh (runs on 1<sup>st</sup> reboot. Install software as 'root' and as 'ubuntu' user

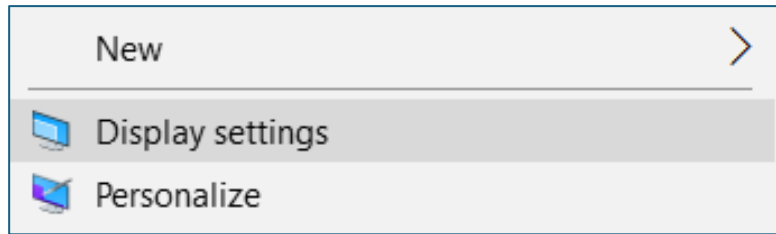
```
1  #!/usr/bin/env bash
2  # =====
3  # postinstall2.sh -- Kiosk Post-Install (Phase 2)
4  # Version: 2.2.2025-10-29
5  # Order of operations (after desktop loads):
6  # ...1) Close Welcome/Tour windows
7  # ...2) Open terminal: tail -f kiosk_postinstall_phase2.log
8  # ...3) Open terminal: top for VMWare processes
9  # ...4) Update sudoers NOPASSWD
10 # ...5) Install VMWare Workstation Pro. (best-effort)
11 # ...6) Extract VM from 7z
12 # ...7) Flatten nested win10/
13 # ...8) Locate .vmx
14 # ...9) Patch .vmx
15 # ...10) Create autostart helper + desktop entries
16 # ...11) Trust desktop entries (gio) in user session
17 # ...12) GNOME lock/idle settings
18 # ...13) Disable Phase 2 service
19 # ...14) Print reboot banner
20 # =====
21 set -Euo pipefail
22
23 # Logging / Tracing
24 # =====
25 LOG=${LOG:-/var/log/kiosk_postinstall_phase2.log}
26 mkdir -p "$(dirname "$LOG")"
27 exec >> (tee -a "$LOG") 2>&1
28 _ts() { date +%F.%T; }
29 log() { printf "[%s] %s\n" "$(_ts)" "$@"; }
30 hr() { printf '\n[%s] =====\n' "$(_ts)"; }
31 PS4='+ $(date +%F.%T) [%BASH_SOURCE##/]:${LINENO}: '
32 # Auto-enable DEBUG
33 # DEBUG=${DEBUG:-1}
34 DEBUG=${DEBUG:-0}
35 [[ "$DEBUG" == 1 ]] && set -x || true
36 trap 'rc=$?; log "[ERROR] rc=$rc at line $LINENO: $BASH_COMMAND"; exit $rc' ERR
37
38 hr; log "==== Phase 2 start (v2.2) ====="
39
40 # Configuration (override via env)
41 # =====
42 SEED_DST=${SEED_DST:-/opt/usb-seed}
43 USER_NAME=${USER_NAME:-ubuntu}
44 USER_HOME=$(getent passwd "$USER_NAME" | cut -d: -f6 || true); : "${USER_HOME}="/home/${USER_NAME}"
45 USER_UID=$(id -u "$USER_NAME")
46
47 VM_ROOT=${VM_ROOT:-"${USER_HOME}/vms/win10"}
48 VMX_PATH_DEFAULT=${VMX_PATH_DEFAULT:-"${VM_ROOT}/win10.vmx"}
49 SPLIT_DIR=${SPLIT_DIR:-"${SEED_DST}/split"}
50 VM_ARCHIVE=${VM_ARCHIVE:-"${SPLIT_DIR}/win10.7z"}
51 VMWARE_BUNDLE_PATH=${VMWARE_BUNDLE_PATH:-"${SEED_DST}/VMware-Workstation-Full-25H2-24995812.x86_64.bundle"}
52 SEVEN_Z=${SEVEN_Z:-/usr/bin/7z}
53 VMRUN_BIN=${VMRUN_BIN:-/usr/bin/vmrun}
54 VMWPRO_BIN=${VMWPRO_BIN:-/usr/bin/vmware}
55 VMPLAYER_BIN=${VMPLAYER_BIN:-/usr/bin/vmplayer}
56 SYS_AUTOSTART_DIR=${SYS_AUTOSTART_DIR:-/etc/xdg/autostart}
57 USR_AUTOSTART_DIR="${USER_HOME}/.config/autostart"
58 USR_APPS_DIR="${USER_HOME}/.local/share/applications"
59 USR_DESKTOP_DIR="${USER_HOME}/Desktop"
60
61
```

This file will likely be changed the most to fit your needs and customizations.

Example:  
Variables have been defined early in the script to allow changes for file names and folders, e.g. 7zip file names for VMWare images.

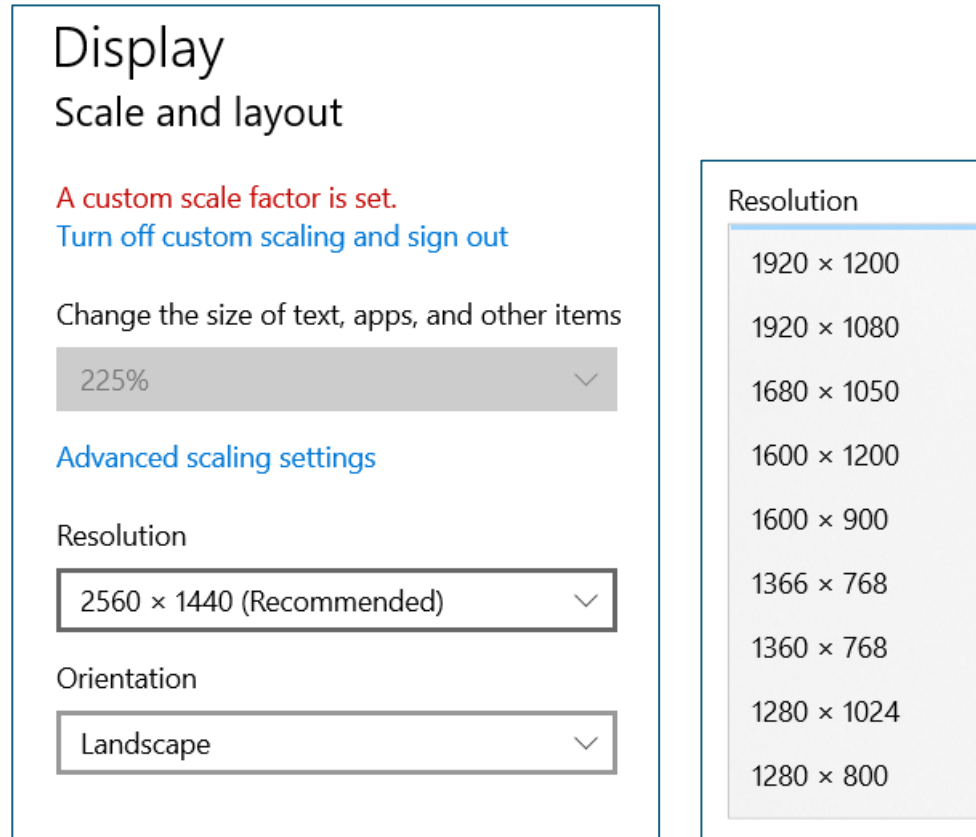


# Resize MS Windows Screen



Resize as needed, depending on the monitors being used.

- Suggest 100% or 125% as needed



# Monitoring Commands

## Phase 1 - Initial Boot - Operating System Install

```
sudo tail -f /var/log/installer/subiquity-server-debug.log
```

```
top -c
```

```
sudo tail -f /target/var/log/kiosk_postinstall_phase1.log
```


## Phase 2 - 1<sup>st</sup> Reboot - Software Install

```
tail -f /var/log/kiosk_postinstall_phase2.log
```

```
top -c
```

```
cat /var/log/kiosk_postinstall_phase1.log
```

Purpose	Log file	Notes
Main installer log	<code>/var/log/installer/subiquity-debug.log</code>	Shows all actions and Python tracebacks.
Installation progress	<code>/var/log/installer/subiquity-server-debug.log</code>	Real-time output during partitioning, package install, user creation, etc.
Autoinstall (cloud-init)	<code>/var/log/cloud-init-output.log</code>	Contains autoinstall status and errors after first boot.
Kernel messages	<code>/var/log/kern.log</code> or <code>dmesg -w</code>	Hardware or driver events.
<b>To follow during install (in a shell, like Ctrl+Alt+F2):</b>		
<pre>bash</pre> <pre>tail -f /var/log/installer/subiquity-debug.log</pre>		

 Copy code