Getting started with Ventoy (v1.09 - 2022-06-14)

Contents:

Chapter 1 - Introduction 5
Ventoy features 6
Chapter 2 - Glossary of terms 7
Chapter 3 - Make a Ventoy multiboot USB drive 9
GPT or MBR? 9
Ventoy partitions 9
Is your USB drive larger than 128GiB? 10
4K sector disks (Advanced Format Storage) 11
Typical Ventoy Legacy\MBR partition arrangement 11
Secure Boot 12
Using the Ventoy installer 14
Beware of fake flash drives! 14
Recommended USB drives 14
Exercise 1: Make a Ventoy drive under Windows 15
Troubleshooting Ventoy installation and updates 15
Exercise 2: Make a Ventoy drive under Linux 17
Re-format Partition 1 19
Updating Ventoy 20
Chapter 4 - Using the Ventoy USB drive 21
Exercise 3: MBR-boot to the Ventoy menu 22
Exercise 4: Add an Ubuntu ISO 24
Exercise 5: Add more Linux ISOs 28
Chapter 5 - Using the Ventoy menu 29
Ventoy menu keys 30
Ventoy Menu Function keys 31
Special Ventoy keys 31
Debug mode 33
Exercise 6: Grub2 mode 35
Troubleshooting summary 38
Chapter 6 - Add more payload files 39

Exercise 7: Add a Windows 10 Install ISO 39
Exercise 8: Add the Parted Magic ISO 41
Exercise 9: Add PassMark's MemTest86 memory test 42
Chapter 7 - Hide some files/folders from Ventoy 45
Chapter 8 - The Ventoy user configuration file (ventoy.json) 47
Ventoy.json syntax rules: 48
JSON Syntax checker 50
VentoyPlugson 50
Multimode 56
Chapter 9 - Ubuntu with persistence 57
Exercise 10: Ubuntu 64-bit ISO with persistence 57
Exercise 11: Kali 64-bit ISO with persistence 65
Chapter 10 - Windows Installs using an XML file 67
Exercise 12: Install Windows 10 Professional onto any computer 69
Exercise 13: Platform specific XML file 72
Chapter 11 - Install Windows 11 to systems with no TPM 74
Exercise 14: Install Windows 11 on old hardware 75
Chapter 12 - Add WinPE ISOs 79
Strelec WinPE 79
Exercise 15: Add the Strelec WinPE ISO 80
Hirens Boot CD WinPE 81
Exercise 16: Add HBCD_PE ISO 81
Other useful WinPE ISOs 82
Chapter 13 - Add AntiVirus ISOs 83
Chapter 14 - Add Chrome OS ISOs (CloudReady and FydeOS) 84
CloudReady (UEFI64 only) 84
FydeOS 84
Chapter 15 - Add Windows .wim files (wimboot plugin) 85
Exercise 17: Add support for booting .wim files 85
Chapter 16 - Add Windows .VHD files (Win VHD plugin) 86
Exercise 18: Add support for .VHD files 86
Chapter 17 - Boot from other drives/partitions/files (grubfm) 87
Exercise 19: Add grubfm and boot ISOs from any drive or partition 87
Boot from Partition 3 using Ventoy 90

Exercise 20: Add StorageCraft Recovery Environment Cross Platform to Partition 3 90
grub2 menu entries for ISO file booting 92
Chapter 18 - Add rEFInd to UEFI-boot from any disk/partition 94
Exercise 21: Add rEFInd.img 94
Chapter 19 - Change the default Ventoy menu theme 97
Exercise 22: Add a 'Seven-of-nine' theme 100
Exercise 23: Random themes 103
Exercise 24: How to make your own Ventoy theme 104
Chapter 20 - Specify the exact menu entries (Image List plugin) 105
Exercise 25: Exclude legacy-only ISO files in the UEFI menu 105
Whitelist menu 108
Chapter 21 - Replace the menu filenames with text (Alias plugin) 109
Chapter 22 - How to display grub2 menu icons (Class plugin) 110
Chapter 23 - Display payload tips to the user (MenuTip plugin) 112
Chapter 24 - Add your own grub2 menus (Extension plugin) 115
Exercise 26: Make a simple ventoy_grub Extended menu 122
Chapter 25 - Auto-select Memdisk mode for some payloads (Auto-memdisk) 124
Chapter 26 - Add passwords (Password plugin) 125
Chapter 27 - Boot payloads from other partitions or disks 127
Using plugins with payload files 127
Exercise 27: Boot a Linux iso+persistence from any partition 127
Chapter 28 - Other Ventoy features 130
Chapter 29 - How to recompile the Ventoy source code 131
Chapter 30 - Add Easy2Boot (Legacy) to Ventoy 132
Chapter 31 - Important BIOS bugs and features that you really need to know about! 135
Chapter 32 - Bootable devices (USB 3 devices are best!) 137
IODD Mini 138
Chapter 33 - So how does UEFI-booting actually work then? 139
UEFI booting 141
EFI Shell 142
Chapter 34 - Secure Boot 143
About UEFI Security (PK, KEK, DB and DBX) 144
Platform Key (PK) 144
Key Exchange Key (KEK) 144

Whitelist Database (DB) 144
Blacklist Database (DBX) 145
Machine Owner Key (MOK) 145
How Secure Boot works 145
Secure Boot and Mok Manager 146
How to disable Secure Boot 147
Chapter 35 - Grub2 configuration file syntax 149
Words 149
Reserved words 149
Quoting 149
Variable expansion 149
Locale strings 150
Comments 150
Simple commands 150
Compound commands 150
Built-in Commands 151
Chapter 36 - grub2 troubleshooting 152
Linux ISO boot issues 153
Theme errors - 'alloc magic is broken' 153
Corrupt screen/bad screen resolution 153
Chapter 37 - Useful links 155

Chapter 1 - Introduction

This eBook explains how to use Ventoy, a USB-based multiboot loader and manager conceived, written, developed and continuously improved by the author 'LongPanda'.

- More eBooks covering grub4dos, grub2\grubfm, Easy2Boot and USB booting are available here. Highly recommended if you want to learn more about USB booting!
- Subscribe to my blog to be informed of any new free revisions of this eBook, useful articles and any new eBooks.

Ventoy is open source software, it is based on grub2 and is freely available under GPL terms on GitHub here.

Ventoy is designed to be installed and run on an external USB drive. Ventoy requires the USB drive to be partitioned in a specific way however, so you cannot add it to an existing USB drive containing one or more operating systems.

This eBook can also be used if you have an Easy2Boot (E2B) multiboot USB drive which includes the 'Ventoy for Easy2Boot' menu system which is extremely similar to the official Ventoy release except the USB drive can be partitioned in a number of different ways. An E2B USB drive can also be temporarily 'switched' to become a Ventoy USB drive by swapping in a partition 2 image. In this way you can easily make the E2B drive appear as an official Ventoy USB drive with 100% compatibility to an 'official' Ventoy USB drive (including ARM\MIPS and Secure booting using the Ventoy method).

You may be interested in some additional eBooks that are available here which are also in this same PDF format.

The Ventoy, E2B and agFM USB menu systems allow you to simply copy over your payload files and then select and boot from that payload file. Currently, Ventoy recognises and supports the following extensions (some additional plugin files are required for some of these):

.ISO
.VHD or .VHDX
.WIM
.EFI
.IMG
.VHD.VTOY (Linux VHD disk images)

Ventoy features

Here are some of the features of Ventoy:

- 100% open source **(Licence)**
- Very simple to use **(Get started)**
- Fast (limited only by the speed of copying iso file)
- Can be installed in USB/Local Disk/SSD/NVMe/SD Card
- Directly boot from ISO/WIM/IMG/VHD(x)/EFI files, no extraction needed
- Can boot payload files from other files and partitions (not just the Ventoy USB drive Partition 1)
- No need to be continuous in disk for ISO/WIM/IMG/VHD(x)/EFI files
- Both MBR and GPT partition style are supported
- x86 Legacy BIOS, IA32 UEFI, x86_64 UEFI, ARM64 UEFI, MIPS64EL UEFI supported
- IA32/x86_64 UEFI Secure Boot supported **Notes**
- Persistence supported **Notes**
- Windows auto installation supported **Notes**
- RHEL7/8/CentOS7/8/SUSE/Ubuntu Server ... auto installation supported **Notes**
- FAT32/exFAT/NTFS/UDF/XFS/Ext2(3)(4) supported for main partition
- ISO files larger than 4GB supported
- Native boot menu style for Legacy & UEFI
- Most type of OS supported, 730+ iso files tested
- Linux vDisk(vhd/vdi/raw...) boot solution **Notes**
- Not only boot but also complete installation process
- Menu dynamically switchable between ListView and TreeView mode **Notes**
- "Ventoy Compatible" concept
- Plugin Framework
- Injection files to runtime enviroment
- Boot configuration file dynamically replacement
- Highly customizable theme and menu style
- USB drive write-protected support
- USB normal use unaffected
- Data non-destructive during version upgrade
- No need to update Ventoy when a new distro is released

You can discuss general Ventoy issues in the Ventoy forums. Ventoy is 100% free - so please make a donation to the developer *LongPanda* if you like Ventoy.

At the time of writing this eBook, Ventoy v1.0.76 is the latest current release.

Chapter 2 - Glossary of terms

AMD64 - Type of Intel x86 CPU architecture which has 64-bit registers extensions. First invented by AMD - a.k.a. a 64-bit Intel architecture CPU or x86-64 or '64-bit'
BIOS - Basic Input/Output System - the firmware inside a device that allows code to perform simple I/O tasks such as write characters to the screen or read sectors or files from a disk.
CSM - Compatibility Support Mode or Module (same as 'MBR BIOS' or 'IBM BIOS' or 'Legacy BIOS'). CSM is essentially the IBM AT-compatible BIOS boot code.
FAT32 and NTFS - two different types of file system. All UEFI BIOSes must be able to access files on a FAT-based file system.
GB and GiB - Gigabyte and Gibibyte - e.g. FAT32 has a 4GiB files size limit (4GiB = 4,294,967,295 bytes = approx. 4.3 GB)
GPT - a new type of disk partitioning which caters for drives larger than 2TB.
grub or **grub2** - grub2 is a bootloader\bootmanager *kernel* used by many Linux based operating systems. It can support x86 Legacy, UEFI64 and UEFI32 systems.
GRUB - Many linux OS's use a complex *script* called GRUB which can help automatically configure a grub2 menu system (e.g. to add another Linux boot entry or modify the behaviour of an existing grub2 boot entry or create a new grub2 menu). Many people (and many documents on the web) confuse GRUB with grub. They are NOT the same thing!
The /boot/grub/grub.cfg file is automatically generated by grub-mkconfig using templates from /etc/grub.d and settings from /etc/default/grub. Any changes you made in in /etc/default/grub file will be reflected in the real grub menu files when you run sudo update-grub.
grub4dos - A Legacy-only open source bootloader/manager used by many solutions including Easy2Boot.
IA32 - x86 Intel Architecture 32-bit (sometimes just called 'x86' or '32-bit')
.ISO - A CD\DVD image file, typically used to *burn* onto writeable CDs\DVDs.
Installer - s/w which installs an OS onto media
Legacy or **Legacy Boot** - IBM AT BIOS compatible booting
LiveCD - A CD\DVD or ISO file which can boot directly to an Operating System. It does not require writeable boot media.
MBR - Master Boot Record - supports Legacy/BIOS/CSM/MBR booting - it's all the same thing!
MBR32 - can be booted by 32-bit and 64-bit CPUs
MBR64 - can only be booted by a 64-bit CPU
NVRAM - Non Volatile Random Access Memory - programmable memory which does not lose its contents when the power is removed. Most modern PCs use EEPROMs
OEM - Original Equipment Manufacturer, such as Dell or HP
Offline OS - the OS to be targeted is not running (e.g. it is just a bunch of files on a disk and you did not boot from it)
Online OS- you have booted from and are running the target system's OS that you want to look at or change
OS – An Operating System such as Linux or Windows or Mac OS X
Payload file - Any bootable file (.ISO, .VHD, .IMG, etc.)
PBR - Partition Boot Record, for MBR disks it is the first sector of a partition (aka **VBR**)
Portable - s/w does not need to be installed into an OS but can be run from external media
Primary MBR partition - a Legacy MBR contains a 64-byte partition table containing four 16-byte Primary partition entries. Logical partition tables are not held in the MBR.
Ptn - shortened form of the word 'Partition' I sometimes use!
SB - **Secure Boot** (signed boot files)
UEFI - supports UEFI booting (in our case x86 UEFI)
UEFI32 - can be booted from a system with a 32-bit UEFI BIOS

UEFI64 - can be booted from a system with a 64-bit UEFI BIOS
WinPE - the Windows Pre-installation Environment (a mini-Windows)
x86 - Type of CPU architecture invented by Intel - often synonymous with 'Intel 32-bit'

Note: Legacy, CSM, IBM BIOS compatible and MBR-booting are all equivalent terms meaning 'non-UEFI' booting.

Chapter 3 - Make a Ventoy multiboot USB drive

The Ventoy menu system is based on grub2 and if you want to make a bootable USB drive you must use the tools provided by the Ventoy website and in the official downloads. When using the Ventoy 'make' utilities provided, the USB drive will be wiped, partitioned and formatted, so *you will lose all data and files* and all partitions.

Ventoy is being constantly improved, so I encourage you to visit the official Ventoy website at https://www.ventoy.net for up-to-date information and new features, etc.

Before we start, you will need to make some decisions about the USB drive that you are going to make, i.e. GPT or MBR partition type, size of partitions, reserved space for use as 3rd partition, file system to use on partition 1, etc.

GPT or MBR?

Ventoy allows you to make a standard BIOS-compatible boot disk or a GPT partitioned disk. You can Legacy and UEFI-boot from a MBR\Legacy disk or a GPT disk, but if your disk is larger than 2TiB and you wish to use more than 2TiB then you will need to choose GPT.

- If you are mainly working on modern systems and modern OS's with **UEFI BIOSes**, I suggest you use **GPT partitions** for your Ventoy disk.
- If you only use a variety of **older systems** then I suggest you use **MBR\Legacy partitions**.
- If want to cover as wide a range of systems as possible, I suggest you **make two Ventoy USB drives** - one GPT (for IBM BIOS+UEFI BIOS) and one MBR (for MBR-only BIOSes). Then delete the \EFI folder from the MBR-only USB drive's second partition for best compatibility and so that it will only MBR-boot.

Some computers (e.g. Lenova Legion, Lenova Yoga, Gigabyte X570 Aorus ULTRA, etc.) may not UEFI-boot unless you use GPT partitioning on the Ventoy USB drive. However, other computers will not MBR\Legacy boot if you use GPT partitions!

Now modern (2019+) PCs no longer support the old MBR\Legacy method of booting from an *internal* drive. Some PCs do not even support MBR\Legacy booting from an external USB drive.

The Ventoy installer makes a small 32MiB FAT partition on Partition 2 (the second partition). This partition contains the Legacy and UEFI boot files used by Ventoy. UEFI-booting simply requires a specifically named .EFI boot file to be present on the USB boot drive in the \EFI\BOOT folder. All UEFI BIOSes *should* be capable of accessing files on a FAT partition (FAT12/FAT16/FAT32, MBR or GPT).

Ventoy partitions

Partition 1: exFAT ('Ventoy') any size
Partition 2: FAT ('VTOYEFI') 32MiB

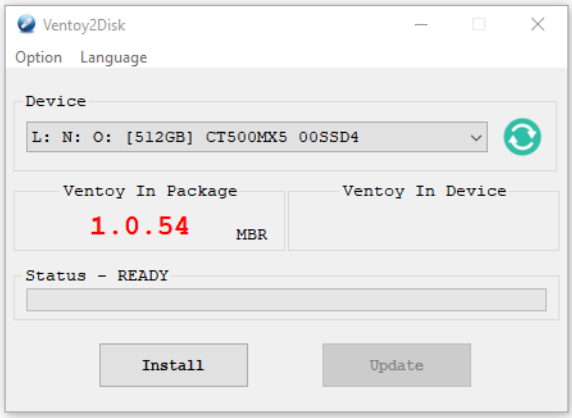
Ventoy assumes that the first partition (Partition 1 - 'Ventoy') will contain your payload files (ISOs, etc.). The second partition usually has a volume name of VTOYEFI (do not change this!) and is usually hidden by Windows; it is always exactly 32MiB is size (FAT).

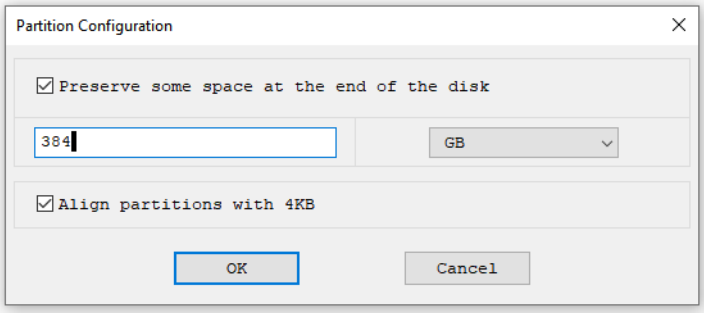
The Ventoy installation utilities will allow you to reserve some space after partition 2 so that you can partition and format more partitions after you have made the Ventoy drive. I recommend that you reserve some space and then later create a third partition on the USB drive using that reserved space. This partition can be used for file storage. You can even store other bootable payload files on it and use 'grubfm' to boot from them.

Once you boot to a Linux ISO under Ventoy, Partition 1 will be 'locked' by Ventoy (unless you use the VTOY_LINUX_REMOUNT option) and you will not be able to write files to it. By creating a third partition, you can use this for file storage under Linux when booting from LiveCD ISO's, etc. and copy files to it, etc.

Is your USB drive larger than 128GiB?

There is a common driver bug in many Legacy BIOSes which means that the BIOS cannot access any sector past 128GiB on a USB drive during booting, so if your USB drive is larger than 128GiB (137GB) and you want to Legacy-boot on a wide range of older systems, then you may want to limit the size of the first partition to 137GB (128GiB). This can be done by using the 'Option - Partition Configuration' tab in the Ventoy2Disk.exe Windows utility. For instance, if you USB drive is 512GiB you could try reserving 384GB of space...





Once you have made the USB drive, you can use a partition tool such as EaseUS Home Partition Master (or GParted if using Linux) to use the unused space to create a third partition. If using MBR partitions, specify a *Primary* partition type (not Logical) for best Windows compatibility.

If your USB drive is smaller than 128GiB or you are only going to use UEFI-booting, then you don't need to reduce the size (however, unless you have a 16GB drive or smaller, I recommend you reserve at least 1GB and then create a 3rd partition afterwards because this will be useful for file storage\transfer).

Note that the first and second partitions that are created by the Ventoy utility or script must NOT be moved or resized afterwards. This is because Ventoy will refuse to run if the size or start point of partition 1 and partition 2 is altered. This means that if you incorrectly calculate the amount of space to preserve, you must use the Ventoy 'make' utility and start all over again!

Note: Partition 2 must start immediately after the end of Partition 1 with not even a single sector gap between the end of Partition 1 and the start of Partition 2 - otherwise Ventoy will fail to recognise the USB drive as being valid. This means resizing partition 1 and then moving partition 2 will usually cause Ventoy to refuse to load!

4K sector disks (Advanced Format Storage)

The Legacy BIOS expects the sectors on a hard disk or USB disk to have exactly 512 bytes of data.

However, these days, most 'spinning rust' disks (hard disks) have sectors which actually hold 4096 bytes (4KiB) of data, but they usually run in 512e compatibility mode which means that we can read and write 512-byte sectors from/to them without a problem. Since the Legacy BIOS expects all sectors of any 'hard disk' or USB drive to be 512-bytes in size, they are fully compatible with the older types of drive which had 512-byte sectors.

Note however that a few USB hard drives use only native 4K sectors. *These drives cannot be legacy booted* and are not supported by DOS/Win98/XP/Vista/Win7 and Windows Server editions before SVR2008 (Microsoft ref.).

Tip: If the manufacturer says that XP\Win7 is not supported or it requires a special driver, then the disk drive is probably a native 4K drive.

Recently, a few *USB flash drives* seem to be appearing which also have 4K sectors and no 512e compatibility mode (this is usually done to improve transfer speeds and benchmark figures - they usually appear as a fixed\local disk and not a removable\USB disk in Explorer. These drives are not suitable for use as multiboot USB drives and will give errors when preparing them with the Ventoy tools.

Ventoy (and Easy2Boot) do not support native 4K USB drives.

Typical Ventoy Legacy\MBR partition arrangement

Note that Partition 1 must start at LBA 2048 (LBA0 is Sector 1 - the first sector on the disk).
Partition 2 must start immediately after Partition 1.

```
Partition 1   SIZE=30537MiB   Type: 07 NTFS\exFAT  *ACTIVE*
START POS   = CYL:0 HD:32 SEC:33       END POS = CYL:461 HD:12 SEC:23
START (LBA) = 2,048 (00000800) SIZE (LBA) = 62,539,776 (03BA4800) [End=62,541,823]

Partition 2   SIZE=32MiB   Type: EF
START POS   = CYL:461 HD:12 SEC:24       END POS = CYL:462 HD:32 SEC:39
START (LBA) = 62,541,824 (03BA5000) SIZE (LBA) = 65,536 (00010000) [End=62,607,359]

P1   Start=2,048 (1,048,576 bytes) End=62,541,823 (32,021,413,376 bytes)
P2   Start=62,541,824 (32,021,413,888 bytes) End=62,607,359 (32,054,967,808 bytes)
```

Secure Boot

UEFI BIOSes can also support Secure Booting. If Secure Boot is enabled in the BIOS configuration settings of the target system, then the security mechanism ensures that only trusted payloads are executed by the UEFI BIOS and any secure boot loader.

Note that it is very easy to Secure Boot to WinPE or Ubuntu, etc. and deliberately infect a system or destroy all data using an automated script. *The Secure Boot protocol is no guarantee that the software you are booting to is benign and safe to use!*

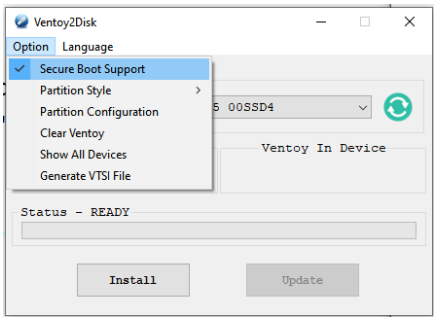
All Windows\WinPE boot files and most Ubuntu and RedHat boot files are signed as well as some other Linux distros.

For instance the .EFI boot file must be signed (by Microsoft). When the UEFI BIOS boots from that file, it will first check that it has been signed and it is 'original'. It is therefore trusted to be *unaltered* since it was originally signed and thus safe to run.

Unfortunately, the grub2 .EFI boot files used by Ventoy are not signed and so they do not support Secure Booting.

However, the UEFI BIOS can read a list of hash keys from the non-volatile RAM (NVRAM) on its motherboard. The BIOS can store a 'white list' of 'good' keys and also a 'black list' of 'bad' keys. So if we add the hash key of the grub2 .EFI boot file to the 'white list' then it will tell the BIOS to trust it.

When you install the boot files onto a Ventoy disk, you can choose the Secure Boot option...



If this option is set, then the first time that you Secure UEFI64-boot to Ventoy, you will usually see a security error message displayed and then you should see the MOK Manager pop-up window where you can add the certificate containing the Key for the grub2 file...

For more details, see the Ventoy web page here.

Once the key has been enrolled, your motherboard will store the key in its NVRAM database (Db) and the next time you Secure UEFI64-boot to Ventoy, you should not see any error messages or see the MOK Manager pop-up window.

Note 1: The MOK Manager utility is not compatible with all UEFI BIOSes and may crash or hang or not load on some systems. In this case you will have to disable Secure Boot in the configuration settings of your BIOS.

Note 2: Easy2Boot uses a different mechanism whereby a signed .EFI boot file is used, however recent Windows updates (KB security updates) have added the hash key to the Dbx 'black list' which blocks it and so you may need to go into the BIOS and clear the Dbx list before you can Secure Boot to 'Ventoy for Easy2Boot'.

Using the Ventoy installer

Before you create the Ventoy boot drive, you need to know the following details (see above):

- Secure Boot (Yes/No)?
- Partition Style (GPT or MBR).
- Amount of space to reserve at the end of the drive for Partition 3.

Beware of fake flash drives!

Make sure you use a known-good USB drive. There are many cheap, fake USB drives about which do not have as much flash memory as they should. For instance, a cheap 128GB flash drive may only contain 8GB of memory (it simply reports that is has a capacity of 128GB). Because Ventoy places its boot files at the very end of Partition 1, a 'fake' USB drive will not boot or will become corrupted when you add large ISO files!

The simplest way to check a USB drive is OK is to fill it at least half-full with large video files (e.g. .mp4 files) and then play the last of those copied video files directly from the USB drive - if some videos will not play then the drive is bad. If you have a Windows system, you can quickly test the USB drive using FakeFlashTest.exe.

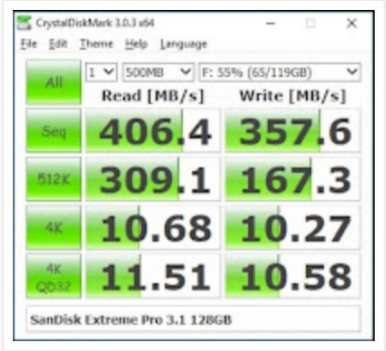
Recommended USB drives

If you use a cheap USB 2.0 drive, you will find that it will take many minutes to copy large files to the USB drive and it will also be slow to use and to boot from.

I highly recommend one of these USB drives for use with Ventoy:

SanDisk Extreme Pro USB SDCZ880 #Amazon ad link (don't buy the 'Go' version!)
Transcend JetFlash 920 #Amazon ad link

Note: If you buy a different USB drive, pay attention to the CrystalDiskMark Random **4KiB** I/O Read/Write benchmark figure. If this is below 1MB/s then avoid it like Covid! Beware of the quoted manufacturers Read/Write speeds which are for *synchronous transfers* and do not give the full picture! Some 'fast' USB Flash drives have very low 4KiB timings and these are *very slow* when used as a boot device or if you try to run Windows To Go or Linux directly from it!

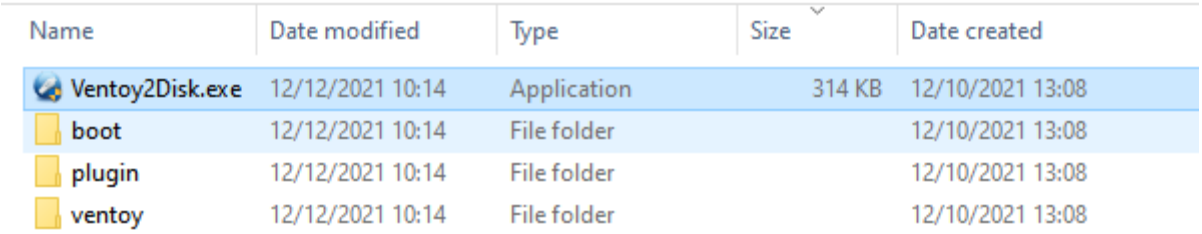


Here are some blog articles which may help you decide.

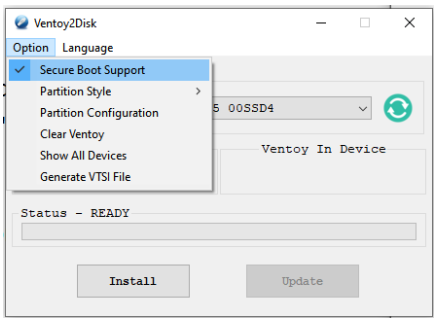
Other devices I recommend include most external USB 3 SSD drives that do not require drivers, the Corsair GTX flash drives (#ad) or a SilverStone (#ad) M.2 enclosure (with added M.2 card).

Exercise 1: Make a Ventoy drive under Windows

1. Download the latest windows.zip version of Ventoy (e.g. ventoy-1.0.64-windows.zip as shown below).
2. Unzip the contents to your Desktop or to a new folder.
3. Run the Ventoy2Disk.exe program from the new folder.



4. Now use the Options tab to set the Secure Boot option if desired, the partition type and any reserved space at the end of the drive...



Note: Some systems, e.g. IdeaPad 300, may not display a UEFI boot option, but instead may list a 'Linpus lite' boot menu option instead of a USB UEFI boot option. Also, if you wish to Secure UEFI-boot on such systems, you may need to choose the GPT Partition Style when you make the USB drive, otherwise it may not UEFI-boot!

Troubleshooting Ventoy installation and updates

Apart from 'fake' USB drives, the most common problem found when making a Ventoy USB drive is that your AntiVirus solution is blocking writes to the MBR and PBR boot code regions of your USB drive. The Ventoy2Disk program creates a log.txt file which you can study for write errors. If possible, temporarily disable or uninstall your AntiVirus software and reboot or reboot in Safe Mode as detailed below.

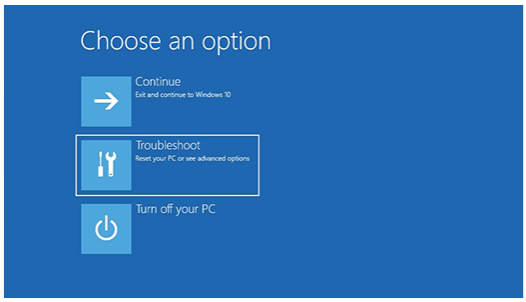
Note: Easy2Boot allows you to add three different Ventoy partition images to the E2B USB drive. This allows you to switch the Ventoy Partition 2 to use either: MBR+non-secure EFI, MBR+Secure EFI (MokManager), MBR+Secure EFI (Kaspersky shim).

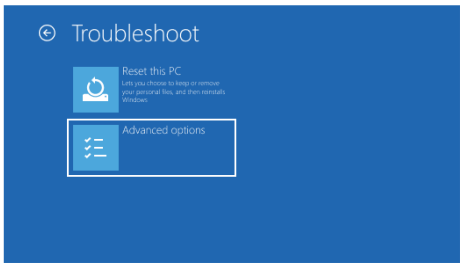
How to boot in Safe Mode

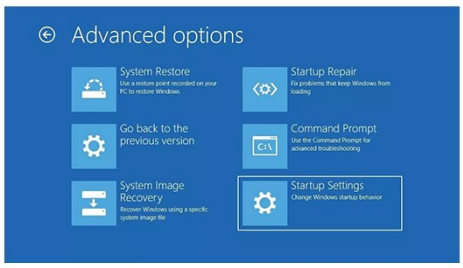
The easiest way to check that your Windows applications are not causing a problem is to boot Windows in Safe Mode:

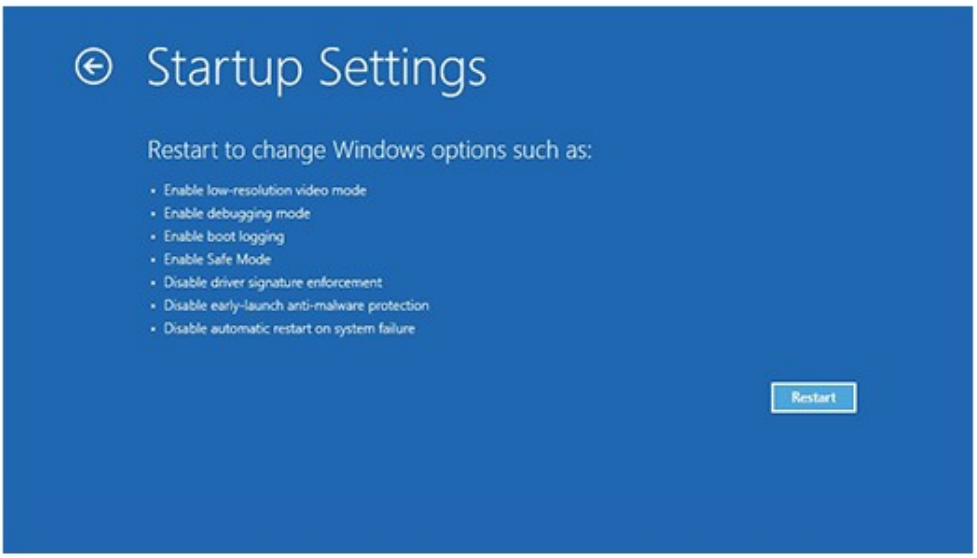
1. Press Windows logo key + **I** on your keyboard to open Settings. If that doesn't work, select the **Start** button, then select **Settings**.
2. Select **Update & Security** > **Recovery**.

   Open Recovery Settings
3. Under **Advanced startup**, select **Restart now**.
4. After your PC restarts to the **Choose an option** screen, select **Troubleshoot** > **Advanced options** > **Startup Settings** > **Restart**. You may be asked to enter your BitLocker recovery key.





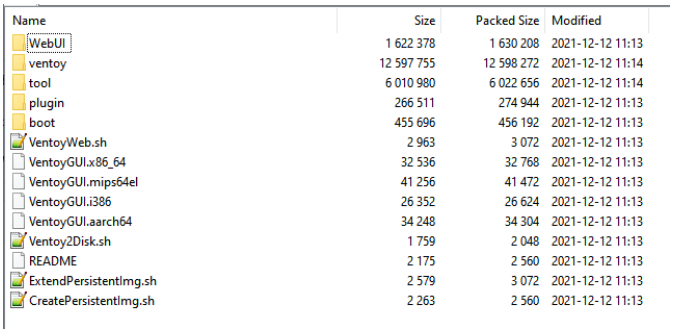




Press F5 at the next screen to boot into Safe Mode with Networking and try the Ventoy utility again.

Exercise 2: Make a Ventoy drive under Linux

If you are a Linux user you will need to download the latest tar.gz file (e.g. ventoy-1.0.64-linux.tar.gz) and extract the contents.



Name	Size	Packed Size	Modified
WebUI	1 622 378	1 630 208	2021-12-12 11:13
ventoy	12 597 755	12 598 272	2021-12-12 11:14
tool	6 010 980	6 022 656	2021-12-12 11:14
plugin	266 511	274 944	2021-12-12 11:13
boot	455 696	456 192	2021-12-12 11:13
VentoyWeb.sh	2 963	3 072	2021-12-12 11:13
VentoyGUI.x86_64	32 536	32 768	2021-12-12 11:13
VentoyGUI.mips64el	41 256	41 472	2021-12-12 11:13
VentoyGUI.i386	26 352	26 624	2021-12-12 11:13
VentoyGUI.aarch64	34 248	34 304	2021-12-12 11:13
Ventoy2Disk.sh	1 759	2 048	2021-12-12 11:13
README	2 175	2 560	2021-12-12 11:13
ExtendPersistentImg.sh	2 579	3 072	2021-12-12 11:13
CreatePersistentImg.sh	2 263	2 560	2021-12-12 11:13

There are two GUI mode programs you can choose from:

Method 1: Linux GUI (GTK/QT) recommended - see table below.
Method 2: Linux GUI (WebUI) VentoyWeb.sh (browser based).

Architecture	GTK2	GTK3	QT5	File	Notes
x86_64	✓	✓	✓	VentoyGUI.x86_64	For most commonly x86 64-bit OS
i386	✓	✓	✓	VentoyGUI.i386	For x86 32-bit OS
arm64		✓	✓	VentoyGUI.aarch64	For ARM64 OS like Phytium/Kunpeng
mips64el		✓	✓	VentoyGUI.mips64el	For Loongson 3A MIPS OS

or you can use command line script 'Ventoy2Disk.sh'...

Ventoy2Disk.sh

1. Download the installation package, like ventoy-x.x.xx-linux.tar.gz and decompress it.
2. Use a command such as df or fdisk -l to view your disk drive device names.
3. Run the shell script `Ventoy2Disk.sh` as root

```
sudo sh Ventoy2Disk.sh { -i | -I | -u } /dev/XXX
```

where XXX is the USB device, for example **/dev/sdb** (do not specify a partition number).

```
Ventoy2Disk.sh CMD [ OPTION ] /dev/sdX
  CMD:
    -i   install ventoy to sdX (fail if disk already installed with ventoy)
    -I   force install ventoy to sdX (no matter installed or not)
    -u   update ventoy in sdX
    -l   list Ventoy information in sdX

  OPTION: (optional)
   -r SIZE_MB  preserve some space at the bottom of the disk (only for
install)
   -s          enable secure boot support (default is disabled)
   -g          use GPT partition style, default is MBR style (only for
install)
   -L          Label of the main partition (default is Ventoy)
```

You can precede the command with 'sudo ' to run as root (depending on your distro) or use 'su -' first.

For more details see the Ventoy website here.

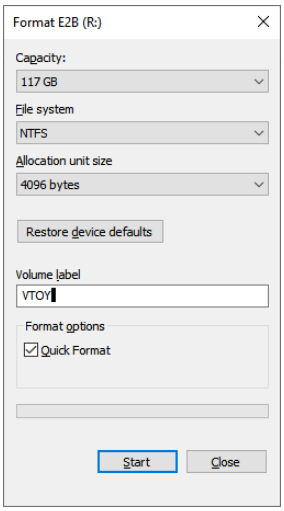
Re-format Partition 1

After the Ventoy USB drive has been made, Partition 1 will usually be formatted as exFAT and will be completely empty.

NTFS is the most compatible if you want Ventoy to boot to Windows VHDs.

If you have a Mac PC or Notebook then exFAT may be the best choice for you, otherwise **I strongly recommend that you re-format Partition 1 to NTFS**.

Under Windows this can be done by right-clicking on the volume letter in Explorer and selecting 'Format...'.

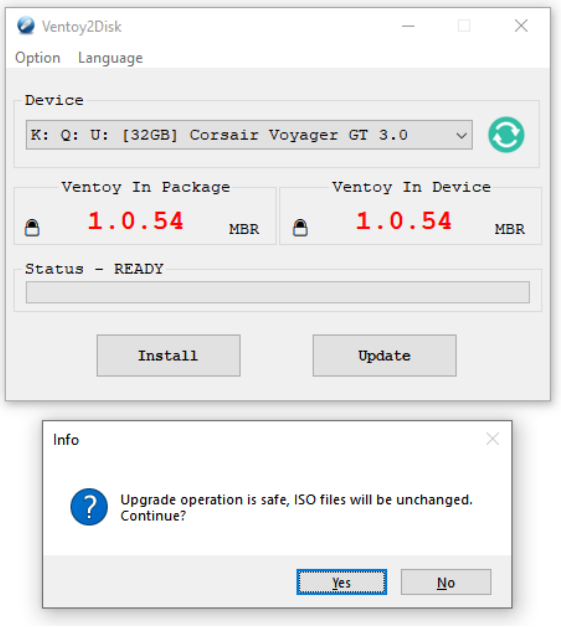


If you are using Linux, you can use GParted or a similar utility to re-format Partition 1. Be careful not to alter the partition size or position in any way. Simple re-format the volume.

Partition 1 can have any Volume name.

Updating Ventoy

The Ventoy creation utility can also update an existing Ventoy disk. The files in Partition 1 will not be touched. The FAT 32MiB Partition 2 (it is type 'EF' and may not be displayed by Windows Explorer) will be deleted and re-written. All other partitions (if any) will not be changed.



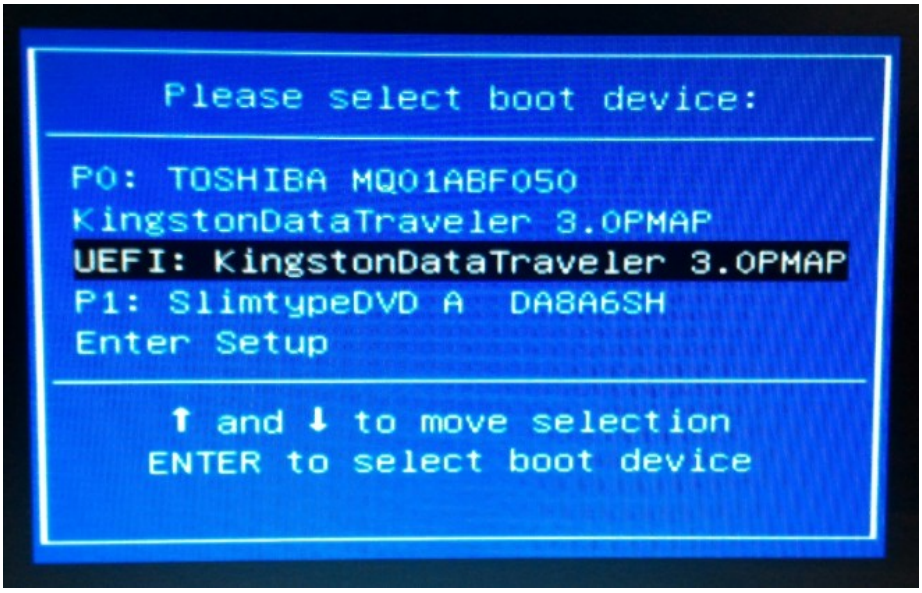
Note that AntiVirus software can block this Update process so you may need to disable it, uninstall it or boot to Safe Mode.

Chapter 4 - Using the Ventoy USB drive

Before you boot from the USB drive, it is important to understand the different boot methods that can be used by an Intel x86-compatible system BIOS:

- **Legacy\MBR booting** - this is the 40-year old Legacy BIOS boot method of loading code directly from a drive's first sector and then executing that code.
- **UEFI32 booting** - an x86-architecture system which contains a 32-bit UEFI BIOS will look for the file \EFI\BOOT\BOOTIA32.EFI on a FAT partition and run it.
- **UEFI64 booting** - an x86-architecture system which contains a 64-bit UEFI BIOS will look for the file \EFI\BOOT\BOOTX64.EFI on a FAT partition and run it.
- **Secure UEFI32 booting** - an x86-architecture system with a 32-bit UEFI BIOS will look for the file \EFI\BOOT\BOOTIA32.EFI on a FAT partition, check its signature is not blacklisted and then run it. Not supported by Ventoy.
- **Secure UEFI64 booting** - an x86-architecture system with a 64-bit UEFI BIOS will look for the file \EFI\BOOT\BOOTX64.EFI on a FAT partition, check its signature is not blacklisted and then run it.

Most BIOSes allow you to press a hotkey (e.g. F2/F8/F10/F12) on power-up or reset to invoke a BIOS Boot Selection Menu from which you can boot to a USB drive.



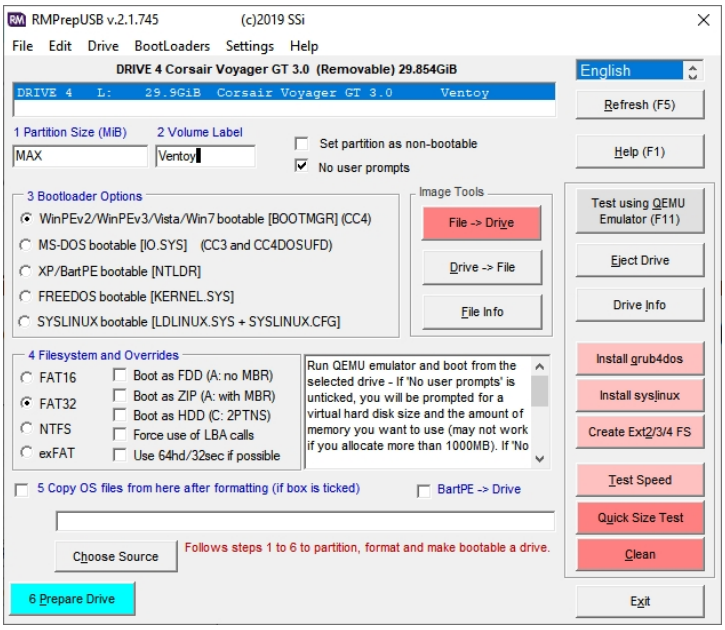
In the screenshot above, there are options to boot from the 'KingstonDataTraveler' USB 3 drive either in Legacy mode or UEFI mode. Note that this BIOS does not tell you if it is a UEFI32 BIOS or a UEFI64 BIOS. It also does not tell you if Secure Boot is enabled or not!

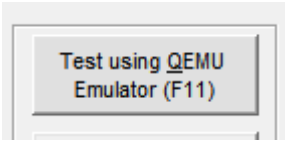
Tip: If your system is having problems booting from a USB drive, then switch off the system - wait 10 seconds and switch it on again. Booting from a USB drive after a 'warm reset' without the BIOS going through it's early POST detection phase, can cause USB boot issues on some older BIOSes.

Exercise 3: MBR-boot to the Ventoy menu

If you have a Windows system, you can *test* that your Ventoy USB drive Legacy-boots to the menu without needing to connect it to another computer or reboot.

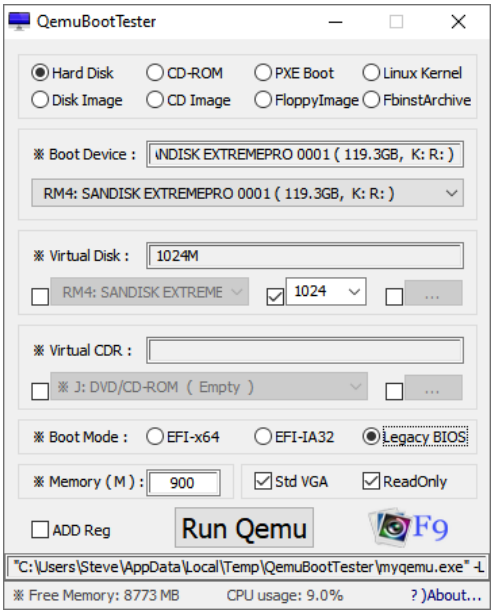
Just download and install RMPrepUSB.exe and press F11:





Note: QEMU will take control of the keyboard. Press *Ctrl+Alt* keys to release control after it has booted (QEMU is quite slow!) and then close the QEMU window when you are finished.

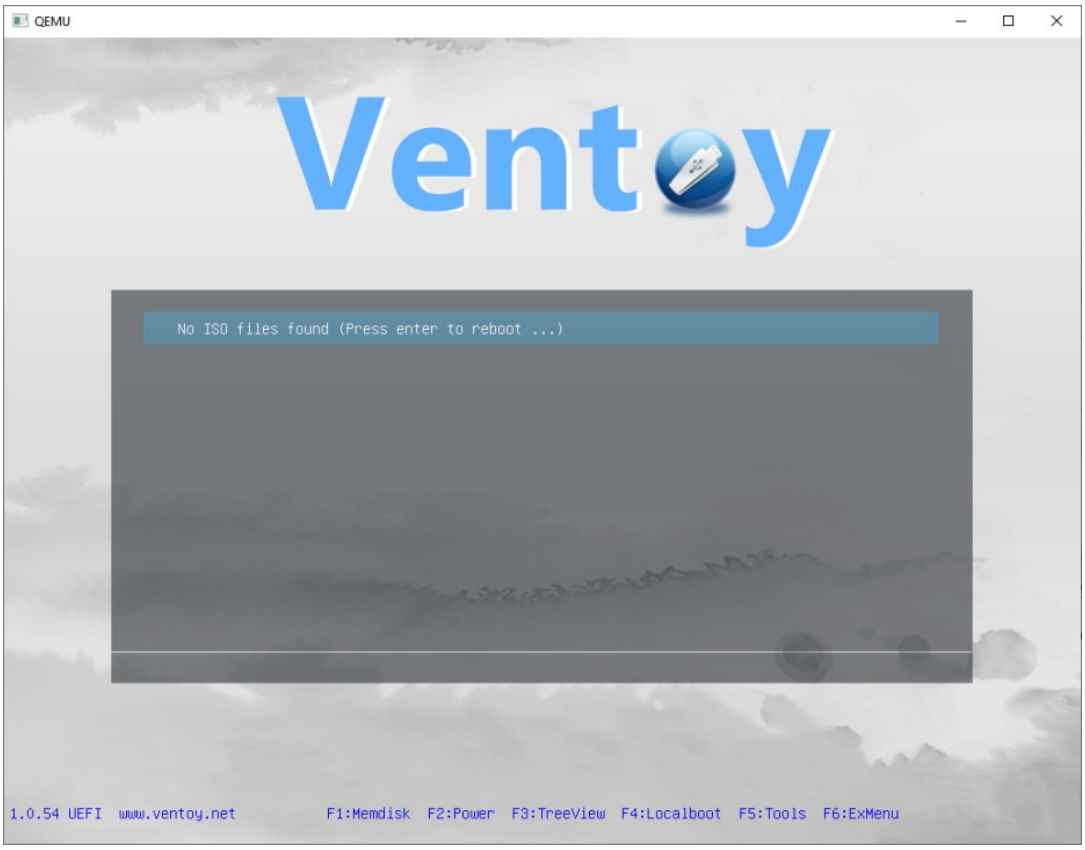
Another alternative is to download **QemuBootTester v7**, which contains the QEMU emulator and it can boot a USB in both Legacy\MBR mode and UEFI64 mode (set RAM to 900). Use Google to find a copy of QemuBootTester v7.



Select '900'M of Memory and either 'Legacy BIOS' or 'EFI-64'.

The above two utilities can be used to test the menu system, etc. but QEMU can be slow and may not work correctly if you select and boot to an ISO from the Ventoy menu.

Here is QEMU in action (with no ISOs files added to the Ventoy USB drive yet):



Note that the Ventoy version number (e.g. 1.0.54 in above screenshot) and the boot mode '**UEFI**' is displayed at the bottom of the menu. If the system has Legacy booted, then the text '**BIOS**' would be displayed instead of 'UEFI'.

The best way to boot from USB drives under Windows is to use Oracle Virtual Box v5 (not v6!) together with the VMUB utility (see here for details). This is the most compatible method and it is much faster than QEMU. It supports Intel x86 Legacy, UEFI32 and UEFI64 (non-secure boot). Note that *UEFI64 Windows 7* booting is not supported by Virtual Box. You can also try VMWare.

If you have a Linux system you can use the version of QEMU which is usually already installed on Ubuntu, e.g.

```
sudo qemu-system-x86_64 -enable-kvm -m 900 -usb /dev/sdb
```

(where sdb is the USB device)

Test using a real system

Now test MBR-booting on a real system. Ensure the BIOS settings are:

- CSM\Legacy = **Enabled**
- Fast Boot = **Disabled**
- UEFI Secure Boot = **Disabled** (may not be required on some systems)

Insert the Ventoy USB drive into the target system's USB port and switch on.

Use your BIOS Boot Selection hotkey (e.g. F8,F11,F12, etc.) and select the USB drive from the BIOS boot menu list (select the EFI or UEFI entry if you want to UEFI-boot). The Ventoy Menu should now be loaded.

If there are no payload files (e.g. ISO files) on Partition 1, then Ventoy will display an error message - 'No ISO files found'.

Exercise 4: Add an Ubuntu ISO

A few Linux ISO files that you can download will only allow you to *install* that version of Linux onto a target disk, however most Linux ISOs are also 'LiveCDs'. This means that when they boot, it loads the Linux kernel and drivers, etc. into memory and the ISO file is mounted as a 'loop' device and you can immediately use it as a Linux computer.

The standard Ubuntu ISO is both a 'LiveCD' and an install ISO. This is typical of many Linux distributions these days.

The Linux kernel will access the files that it needs via the 'loop' device which will access the files inside the ISO file on the USB drive (or CD/DVD). Note that if Ventoy's 'MemDisk' F1 option is used (or a kernel parameter such as 'toram' is used) then the files on the ISO/CD/DVD will be copied into memory first - thus you can remove the USB/CD/DVD drive after it has booted to Linux (the system needs to have a large amount of RAM for this - or else the ISO must be quite small).

Note that when you boot from a 'LiveCD', any changes that you make to the Desktop or file system will all be lost as soon as you shutdown or reboot. *The changes will not be persistent* when using a LiveCD ISO.

You should also be aware that the ISO may support Legacy\BIOS booting only, or UEFI64 only. Some modern BIOSes may not boot from external media by default - check your motherboard/notebook manual.

Follow the steps below to boot from an Ubuntu ISO file on a real system.

1. Download the Ubuntu Desktop ISO file from here.

2. By default, Ventoy will enumerate all the ISO files that are on Partition 1 and will list them in the menu. I suggest that you place the Ubuntu ISO file in a new folder on your USB drive. Since you can then put other files on the same USB drive (e.g. your own personal files, utilities, etc.). I suggest you use a folder structure like this:

\ISO\ANTIVIRUS
\ISO\LINUX
\ISO\BACKUP
\ISO\UTILITIES
\ISO\WINDOWS INSTALL
etc.

(Easy2Boot uses a similar folder scheme, except _ISO is used instead of \ISO).

So in this case, first make a \ISO folder and then a \ISO\LINUX folder and then copy the Ubuntu ISO to the new **\ISO\LINUX** folder.

Later we will create a folder on the first Partition called \ventoy which is used to hold configuration files and Ventoy 'plugin' files, etc., so *do not create a folder named \ventoy* at this stage.

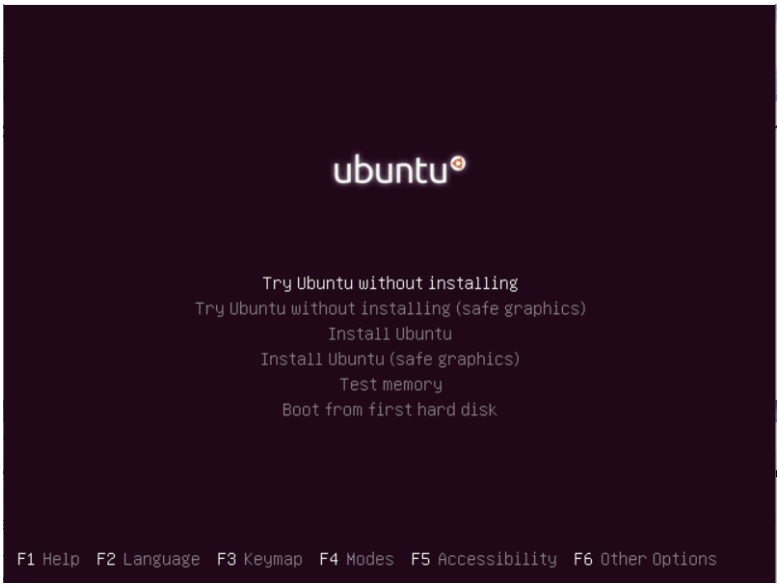
Instead of \ISO, you could use **\BOOT FILES** or **\BOOT** perhaps, since in addition to ISO files, we will also later-on add other types of bootable files such as WIM, IMG, VHD, EFI, etc.? e.g. \BOOT\LINUX\UBUNTU.

3. Now Legacy or UEFI-boot to the Ventoy menu using a real system (or VBox+VMUB) and select the Ubuntu ISO which should now be listed in the menu.

< Note: Legacy mode boot = 'BIOS'.

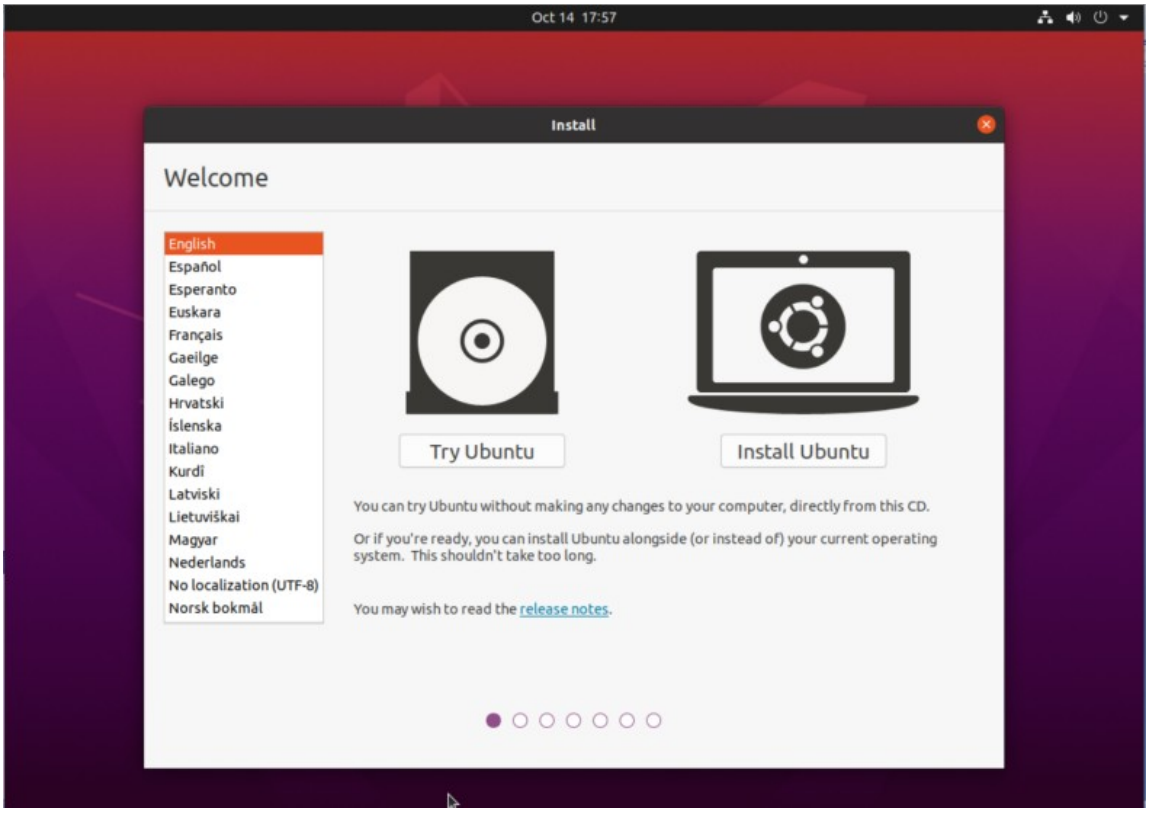
You can now use the cursor keys to select the ISO (if there is more than one) and then press ENTER to boot from it.

Ventoy does not always offer you the same boot menu that is actually embedded within the ISO. The ISO's embedded menu will usually allow you to choose different options and you can often modify the menu parameters (e.g. by pressing 'e' for edit) before you boot to the Linux kernel.

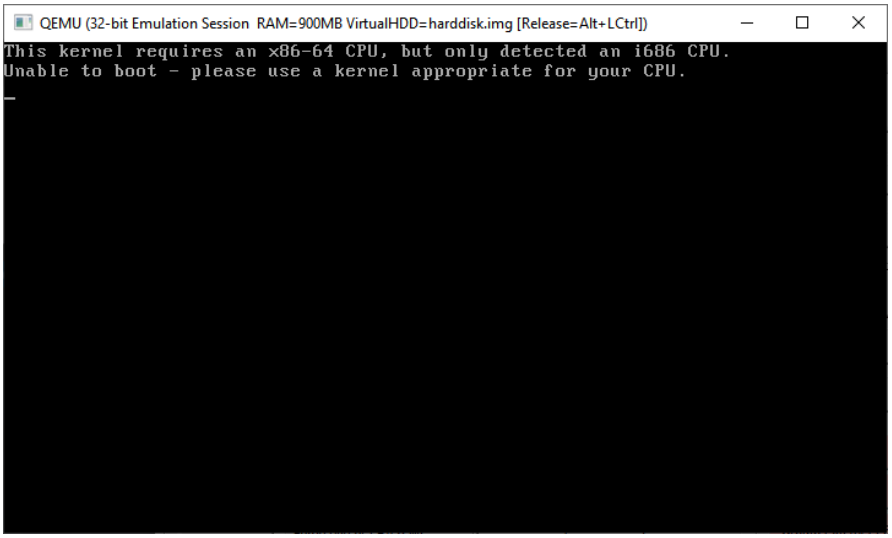


This embedded Ubuntu menu is *not* always displayed by Ventoy.

When you boot to the Ubuntu ISO, you should see a Desktop as shown below...



Note: Because the version of QEMU used with RMPrepUSB.exe uses a 32-bit CPU emulator, you will see a message similar to the one below if you try to actually boot to a 64-bit Linux ISO with the QEMU emulated 32-bit CPU.



< *'that dog don't hunt!'*

Exercise 5: Add more Linux ISOs

Simply repeat the steps above to add more Linux ISOs and then try them out.

To download some of these files under Windows, you may have to *right-click on the link* in your browser and choose the *'Save link as'* to download the file and then choose *'Keep'* if you get a security warning from Windows Defender, etc.

Ventoy supports a wide range of ISOs, the currently supported list can be checked here.

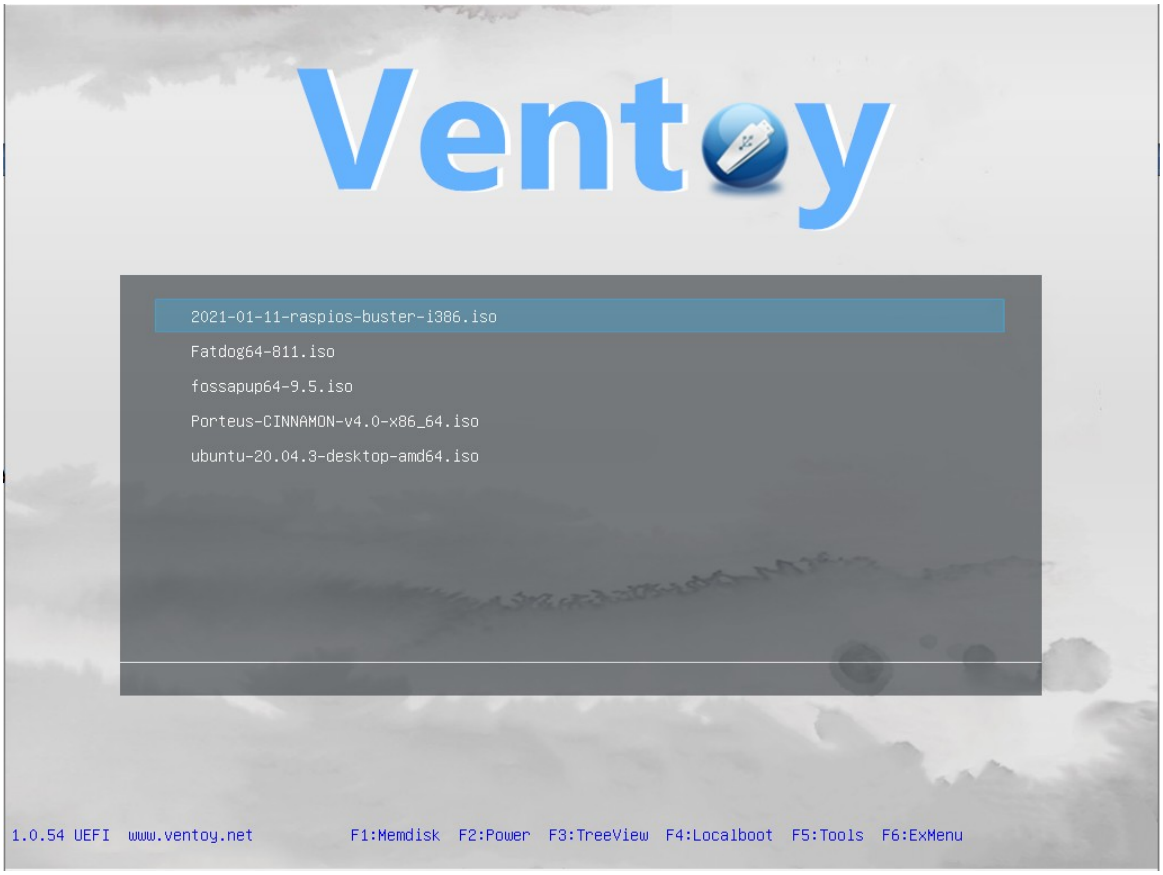
Here are some you might like to try (links may become outdated):

Nitrux - for beginners
Zorin OS - best for Windows users
Pop!_OS - best for gaming
Kodachi - for privacy and security
Rescatux - for repair and rescue
Parrot - for forensics/penetration testing
OpenMediaVault - for installing to make a NAS PC
Porteus - mainly Legacy only - designed to be run from USB with persistent storage by default (check list here)
Manjaro - Arch-based Linux
Puppy Linux - 32-bit and 64-bit versions available, multiple flavours, lightweight
FatDog - similar to Puppy but can save changes to a file on the USB drive on exit
Arch Linux - for power users
Solus - for developers
Raspberry Pi Desktop (Debian Duster) - put your finger in the Pi! (X86 PC and Mac)
Kali Linux - for forensics/penetration testing
Elementary OS - (tip: use Custom - 0 to get for free) - attractive Desktop
Linux Mint - Ubuntu-based, popular Linux OS
Fedora - bleeding edge, regularly updated
RedHat Enterprise Linux - server-based free version of commercial distro (large)
Tails - safe, secure browser
Ikki Boot - large ISO containing all the most useful LiveCDs you could ever want, including Slitaz, GParted, Clonezilla, grubfm and a lot of tools like TestDisk, SystemRescue, MemTest86+, Offline NT password reset, Puppy Linux, Darik's Boot and Nuke and Super Grub Disk. UEFI and Legacy boot.

Note: Ventoy may 'lock' the first partition once it has booted to Linux, so you may not be able to access or save files to this partition from the Live Desktop. If you wish to save files to the USB drive then you should create a third partition on the drive of at least 1GB in size. For instance, FatDog will offer to create a *save file* (like a persistence file) the first time you shut it down, so at that point you can select Partition 3 to save the changes file because Partition 1 will be inaccessible and Partition 2 is too small.

Chapter 5 - Using the Ventoy menu

Your Ventoy menu may contain one or more ISO files by now and the Ventoy menu screen might perhaps look something like that shown below:



Note that all the files will be sorted into alphanumeric order (even if they were placed in different folders).

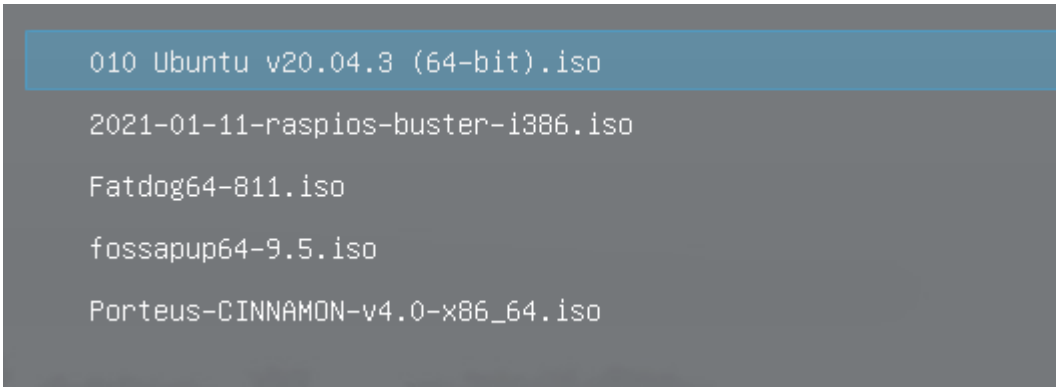
An easy way to change the order is to rename each ISO filename with a three digit number prefix - e.g.

010 ubuntu-20.04.3-desktop-amd64.iso
020 Fatdog64-811.iso
030 fossapup64-9.5.iso

Tip: I use three digits in intervals of 10 *because it makes it easy to add new files at a later date* (e.g. 012, 014, 018, etc.) without needing to renumber all the other ISO files at the same time! You can also quickly sort the files into the same order using Windows Explorer or the Linux File manager.

You can change the filename (there is nothing 'magic' about the filenames!), e.g.

'010 Ubuntu v20.04.3 (64-bit)' as shown below...



You can use spaces in filenames, but it is best avoid 'special' symbols such as $ or % if possible.
Tip: You will later learn how to display any text you like for any file using an 'Alias' key.

Ventoy only lists files with certain specific file extensions, so do not change the file extension.

Note: Some versions of Porteus have UEFI boot issues (see here for details).

Ventoy menu keys

The Ventoy grub2 menu will respond to the cursor up/down keys as well as the Page Up/Page Down/Home/End keys. A mouse/trackpad is not supported in Ventoy.

grub2 also allows the use of these keys:

Grub uses Emacs-like key bindings:

- Ctrl + B = Left Ctrl + F = Right (mnemonic: backward/forward)
- Ctrl + P = Up Ctrl + N = Down (mnemonic: up/down)
- Ctrl + A = Home Ctrl + E = End (mnemonic: A=beginning/end)

More details on grub2 keys can be found here.

The ESC key will normally take you back to the previous screen or back to the menu from the grub2 command line.

If a menu entry is too long, you can scroll the text sideways on the screen using Ctrl+F and Ctrl+B.

Pressing the 'e' for 'edit' key will display the grub2 code used for that particular menu entry. You can then use the grub2 edit feature to temporarily change the menu code and press F10 to run it with your changes (your changes will not be permanent).

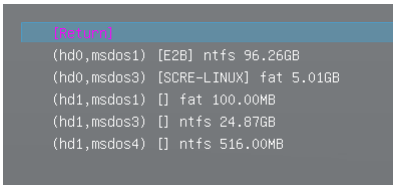
Ventoy Menu Function keys

h = help menu

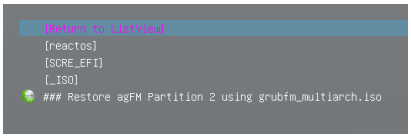
Try out these Function keys (if using a laptop, you may need to also press the Fn key as well as the F-keys in the top row of the laptop keyboard).

F1: Load into Memory - Ventoy will load the entire ISO/payload file into memory before booting from it. Try this if your payload does not boot successfully in normal mode. With some Linux ISOs you may be able to remove the USB drive once it has booted too! _VTMEMDISK

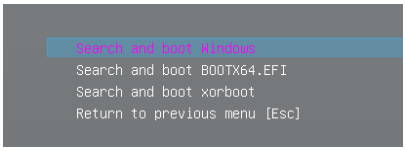
F2: Browse - This menu allows you to browse bootable files on any disk\partition in the system.



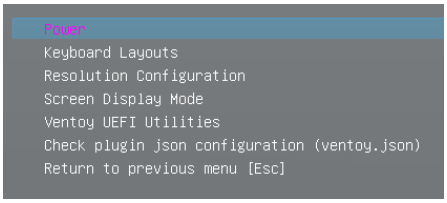
F3: Tree View - Change the menu from List View to Tree View mode. In Tree View, Ventoy will list files and folders in the current folder only. You can select a different folder and you will only see the payload files within that folder.



F4: LocalBoot - This menu allows you to boot from a disk containing Windows or grub4dos (/grldr file) or from the first partition of local disks 1-3 in the system. Press ESC to exit the F4 menu.



F5: Tools - Power (reboot/halt), pick a different Keyboard layout, screen resolution, display mode (text or graphics). You can also syntax check each section of your *\ventoy\ventoy.json configuration file* (checks for case, path, etc. as well as bad syntax).



F6: ExMenu - runs the users external grub2 menu - **\ventoy\ventoy_grub.cfg**. By using an external menu you can add more functionality to the Ventoy menu including designing your own custom submenu system and adding grub2 menus.

Special Ventoy keys

F7: Text\Graphics mode toggle - sometimes when you UEFI-boot from an ISO, you may see a graphics display issue caused by an unsuitable resolution being used for the graphics adaptor or display screen. Switching to Text Mode before selecting the ISO may fix this. Also, some systems react very slowly to the Up/Down cursor

keys when in graphics mode - using Text mode can greatly speed up the Ventoy menu scroll speed on these systems.

h (Ctrl+h v1.0.56)**:** Help information (v1.0.57+)

Ctrl+i: Compatible mode - if you have problems booting any ISO, try Compatible mode.

Ctrl+w: Wimboot mode - if you have trouble booting to an official Windows ISO, try using Wimboot mode. _VTWIMBOOT

Ctrl+r: Grub2 mode - use if you have trouble booting a Linux ISO (use if 'out of memory' error). _VTGRUB2

Ctrl+u: Load ISO UEFI driver (for debug only).

m (Ctrl+m v1.0.56)**:** Calculate the md5/sha1/sha256/sha512 hash of the selected file. You can also add a file containing the expected hash (e.g. Ubuntux64.iso and Ubuntux64.md5 - see here for details) (v1.0.57+).

c - go to grub2 command console (ESC to return).

e - edit current grub2 menu entry (edits are not permanent).

Ventoy 1.0.76 and later versions also allow you to modify the name of an ISO file with a suffix so that it will use one of these modes without needing to use a ctrl+ key chord first:

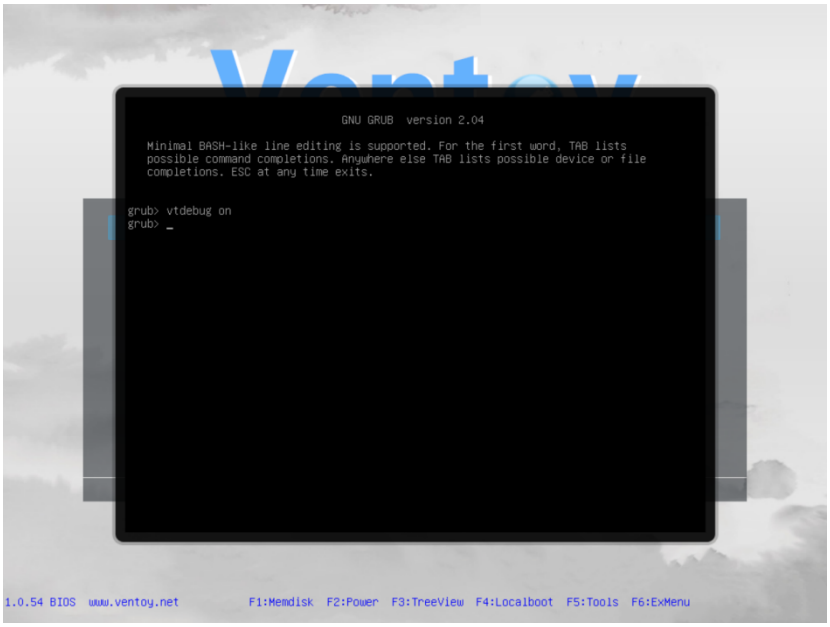
1. If the ISO file name ends with **_VTMEMDISK**, Ventoy will automatically use Memdisk mode.
For example: kolibri_VTMEMDISK.iso
2. If the ISO file name ends with **_VTGRUB2**, Ventoy will automatically use GRUB2 mode.
For example: ubuntu-22.04-desktop-amd64_VTGRUB2.iso
3. If the ISO file name ends with **_VTWIMBOOT**, Ventoy will automatically use WIMBOOT mode.
For example: win11_english_x64_VTWIMBOOT.iso

These three special suffixes are not case sensitive.

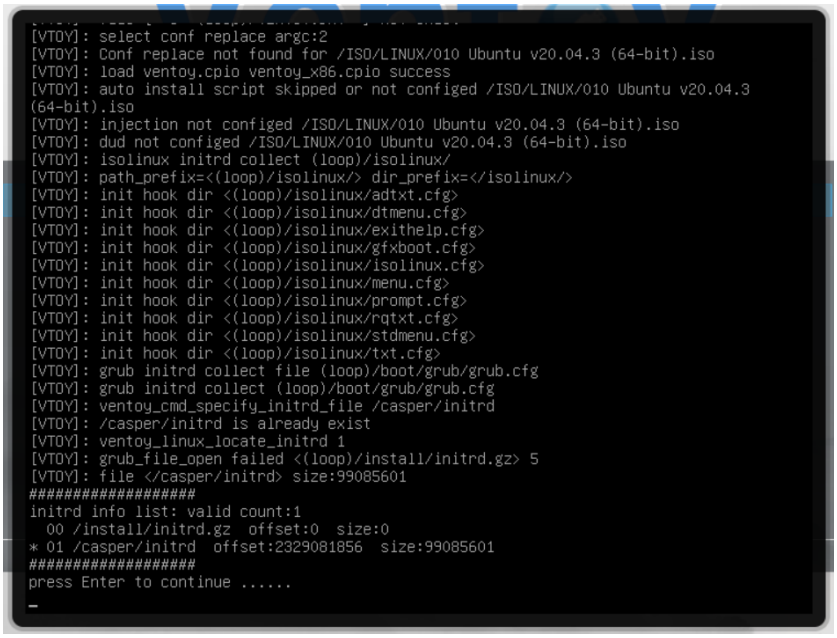
Some ISO files will only work with ctrl+r - e.g. Austrumi, 4MLinux, ubuntudde,

Debug mode

To debug Ventoy – Press **c** in the Ventoy menu system to enter the grub2 command line – then type **vtdebug on** and press **Enter** and then press **ESC** to return to the Ventoy menu. Ventoy debug messages will then be displayed when you select and run a payload file.



Screen output from Ubuntu ISO + Ventoy's debug mode...

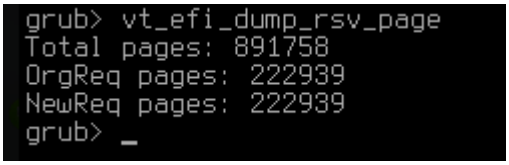


If there is not enough free memory below 1MB then use ctrl+r grub2 mode before selecting the ISO (common issue with HP computers even if they have over 4GB RAM).

The grub2 console command **vt_efi_dump_rsv_page** command dumps the number of free pages of memory under 4GB.

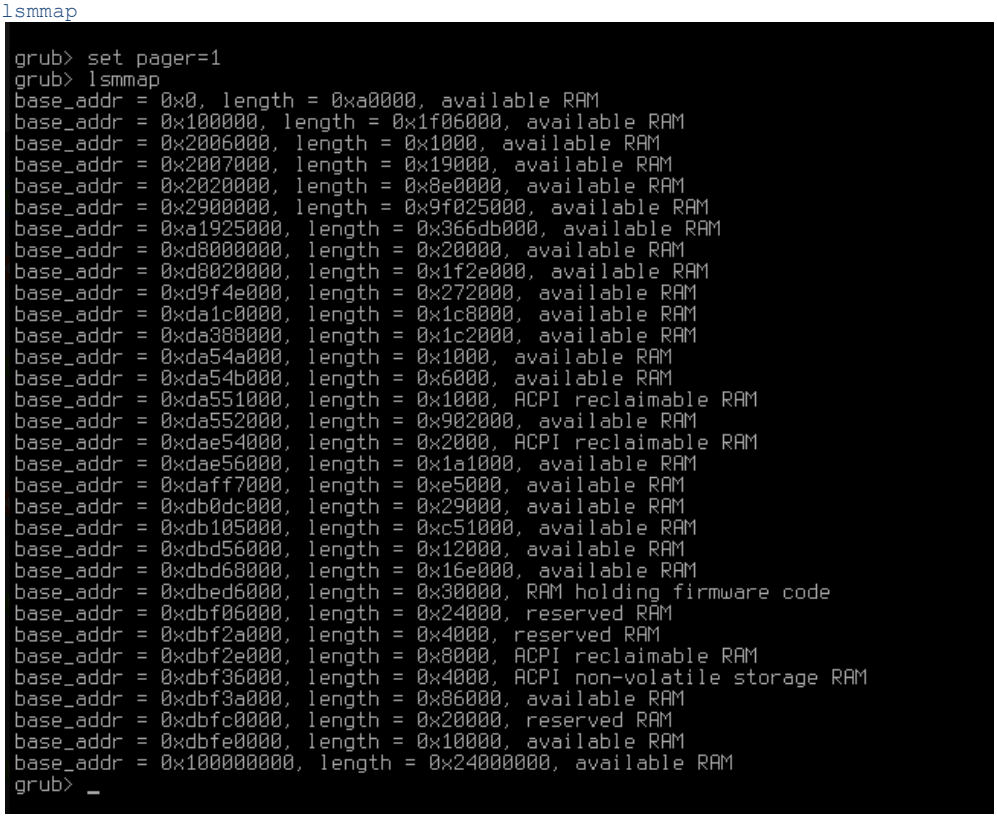
Total pages: 79533
OrgReq pages: 19883
NewReq pages: 19883

A value of 79533 is 79533 x 4KiB = 310MiB (very small!)
A value of 182173 is 182173 x 4KiB = 711MiB (not great!)
A value of 891758 is 891758 x 4KiB = 3,483MiB (typical on 4GiB+ system or VM)
It's the BIOS that decides how much free memory can be used under the 4GB space address.



Also try:

```
set pager=1
lsmmap
```

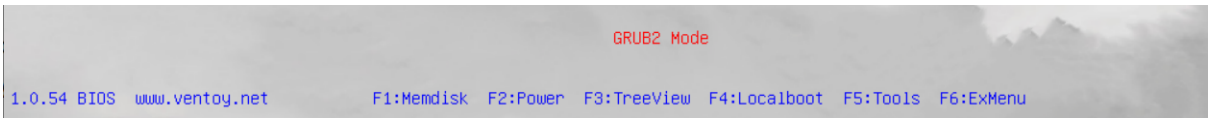


This allows you to see the areas of free memory (usable RAM) in the system memory map.

Exercise 6: Grub2 mode

1. Legacy/MBR Boot in normal mode to the Ventoy menu (not UEFI) and select the Ubuntu ISO file. Press ENTER to boot from it. Note that if you do not UEFI-boot, it boots straight to Ubuntu.

2. Re-boot to the Ventoy menu and press Ctrl+R for grub2 mode. You should see 'GRUB2 Mode' displayed in red above the menu help tips...



3. Now select the Ubuntu ISO again and press ENTER.

4. You will see the ISO's own internal grub2 menu.

Note: Ventoy 1.0.76+ allows you to modify the ISO name and add _VTGRUB2 to the end of the name so that you do not need to press Ctrl+R first.

5. Press the SPACEBAR quickly to stop the 5 second timeout.

6. Now press 'e' for grub2 edit.

7. You can now directly edit the Ubuntu grub2 menu commands. Typically you may want to add a new kernel parameter just before the **---** text and remove the **splash** and **quiet** words so you can see all kernel messages (see red arrow above).

Tip: Pressing ESC or F1 during booting often causes Linux to show the initial boot messages.

In the example below **edd=off** disables the BIOS Enhanced Disk Drive service and the words **quiet** and **splash** have been removed. Note that any kernel parameters placed *after* the --- or -- dashes will be ignored as they are passed to init and not to the kernel. I could have also added the parameter **debug** to get more messages during boot up of this ISO.

Tip: If you have problems booting, you can try typing **dmesg** at the Linux terminal command prompt and scroll through all the diagnostic messages.

Each Linux distro has a wide variety of different kernel parameters - some may only be valid in Legacy mode (and others are only valid in UEFI mode). Even different 'flavours' of the same Linux build may support different kernel parameters, so you will need to look up the specific parameters for a specific build and version of the Linux that you are booting. A general list of parameters for 'Ubuntu' can be found here.

Troubleshooting summary

If you have problems booting to a particular ISO or other payload file:

- Test using the original unchanged name of the ISO first. Once it is working you can change the filename to something prettier.
- Check both Legacy and UEFI booting works OK (the Ventoy menu tells you how you have booted, either as 'UEFI' or 'BIOS')
- Try a different make\model of computer to see if the issue is model\BIOS dependent.
- If the problem is only in UEFI-mode, try using F7 text menu mode before selecting the ISO file. Also try changing the Ventoy screen resolution to 800x600 first or perhaps something higher than 1024x768.
- Try F1, Ctrl+i, Ctrl+r or Ctrl+w (one at a time) before selecting the ISO.
- Use '`vtdebug on`' (see above) to see any error messages from Ventoy when you select an ISO.
- Add the 'debug' parameter to the kernel parameters ('linux' grub2 command - see screenshot above) - this may give a clue as to where the problem lies (e.g. graphics or sound driver, etc.).
- If the problem is model-specific or specific to a particular Linux distribution, you may need to add a kernel parameter (for instance when you have an nVidia graphics card). You will need to search using Bing\Google for other users who have had a similar issue to find a solution!
- If you are Secure Booting and see some sort of '**error: bad signature**' or '**kernel.dat not found**' error (but only when using Secure Boot) then press Ctrl+r, select the ISO in the Ventoy menu and then in the next grub2 Linux (e.g. Kaspersky or whatever you selected) boot menu (not when in the Ventoy menu), press 'c' and then type **set check_signatures=no** in the grub2 command console and then press 'ESC' key to get back to the Linux boot menu.

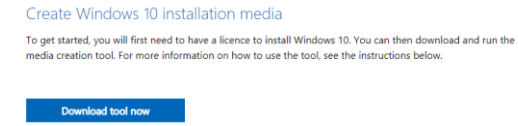
Chapter 6 - Add more payload files

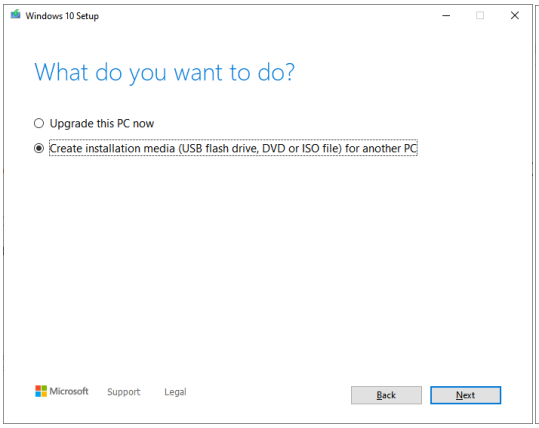
Ventoy supports booting to Windows 7/8/10/11 Install ISO files (but not Win98/Me/XP/2003). Using these Microsoft ISOs you can install Windows onto another disk or repair an existing Windows installation.

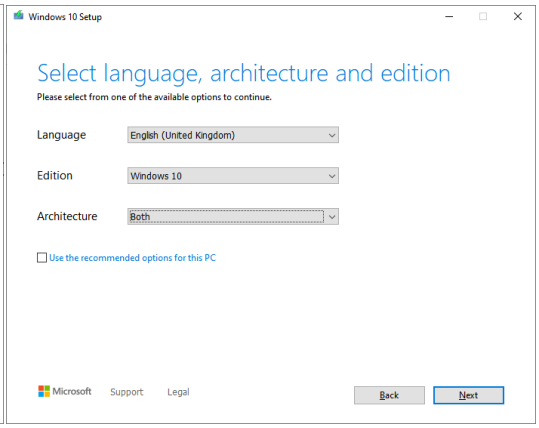
Exercise 7: Add a Windows 10 Install ISO

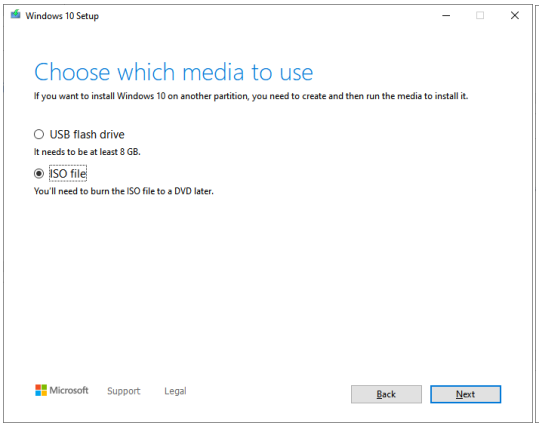
1. Go to the Microsoft official web page here.

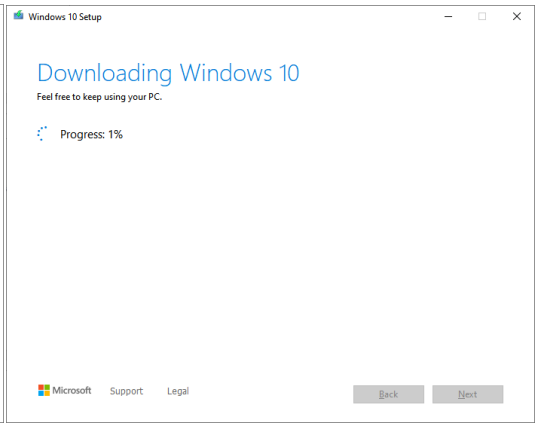
If you are using a Windows OS then download the 'Media Creation Tool' which is currently an .exe file and run the executable...





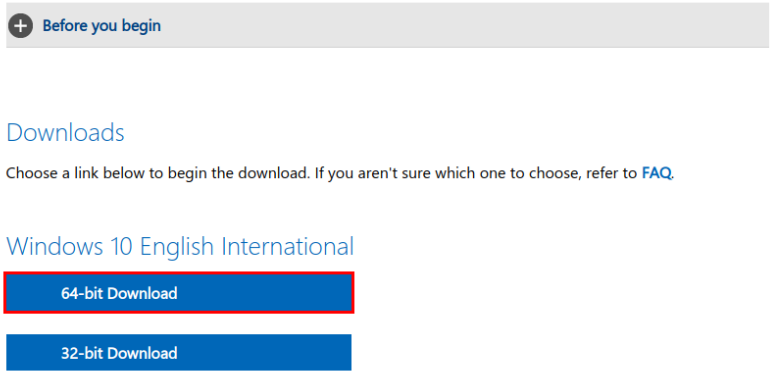






Note: If the download Progress percentage count pauses, make sure that you dab the cursor onto the window so that the utility is running in the 'foreground'.

If you are using a browser under **Linux**, then you can directly download the ISO (no tool is provided).



Note that a Windows Product Language of 'English' really means *'American* English' (a corrupted/modified version of true 'English' ;-), whereas 'English International' or 'English (United Kingdom)' means proper English (the English race and the English language actually originated from England) plus all other English speaking ~~colonies~~ countries. If you just choose the Microsoft Windows 'English' ISO then you will not be able to install Windows in 'UK' English or any the versions of English used by many other countries (it is only for the USA).

2. Copy the downloaded ISO file to your Ventoy USB drive. For instance, you can copy it to the \ISO\WINDOWS INSTALL\ folder.

3. Now you should be able to Legacy\MBR *and* UEFI64-boot to Ventoy and install Windows 10 using that ISO file.

Note that unlike many other operating systems, Windows Setup has a strict requirement in that if you **Legacy boot**, then Setup will only install Windows to a disk containing **MBR** partitions. If you **UEFI**-boot then Windows Setup will only install to a disk with **GPT** partitions. If the disk is unpartitioned (clean/raw) then Windows Setup will automatically use the correct partition scheme.

Tip: Some older versions of Windows Setup will not 'clean' a drive fully if you 'Delete' all partitions and then auto-install on a 'raw' drive. For instance Windows Setup may refuse to create MBR partitions if you Legacy-boot to Setup and you then try to install Windows onto a disk that previously had GPT partitions. In this case, you may need to run DISKPART to *'convert'* the disk as follows:

1. After booting from the Windows ISO and in Windows Setup, press the two keys SHIFT+F10 to open a command prompt.

2. Type the following commands in the new cmd window:

```
DISKPART
LIST DISK
LIST VOL
SEL DISK n
CLEAN
CONVERT MBR
EXIT
```

Where **n** is the number of the disk you want to completely wipe and install Windows to.

If you have a similar issue when you UEFI-boot to Windows Setup, then use 'CONVERT GPT' instead of 'CONVERT MBR'.

Exercise 8: Add the Parted Magic ISO

This is a useful, well loved Linux-based ISO which can be used for preparing disks.

1. Download the ISO from here (requires a subscription fee which can be cancelled after downloading) or an older version from here or search the internet for a 'pmagic' iso.

2. Copy the ISO file to a suitable folder on the Ventoy USB disk such as \ISO\UTILITIES.

3. Try it, but be careful not to destroy any partitions in your system!

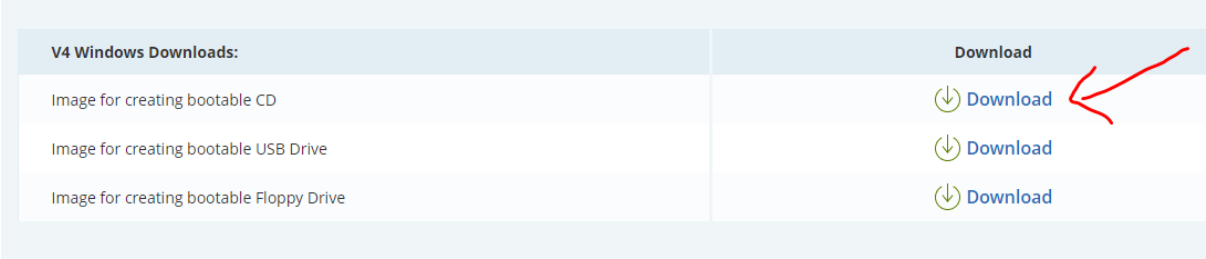
Exercise 9: Add PassMark's MemTest86 memory test

MemTest86 is a well known memory test which can be used as an alternative to the memory test on the Microsoft Windows Install ISOs (which is also good).

*P.S. A useful thing to know is that the MemTest86 EFI files are actually SecureBoot **signed** and so a FAT32 USB drive that just contains MemTest86 should always boot on any Secure Boot-enabled system. In this case however, we always boot to Ventoy first...*

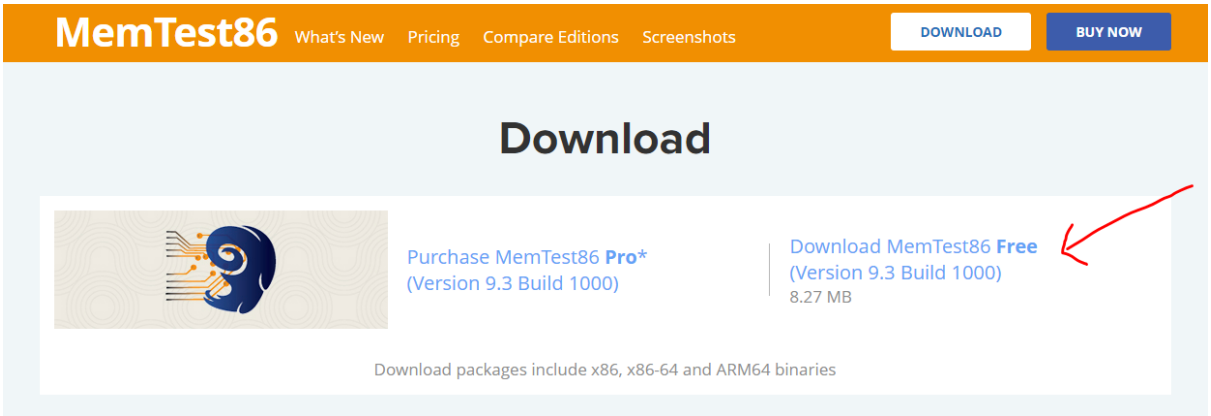
1. Visit the download page here.

2. Download the 'Old' v4 ISO file (this is a Legacy-only version of the memory test)

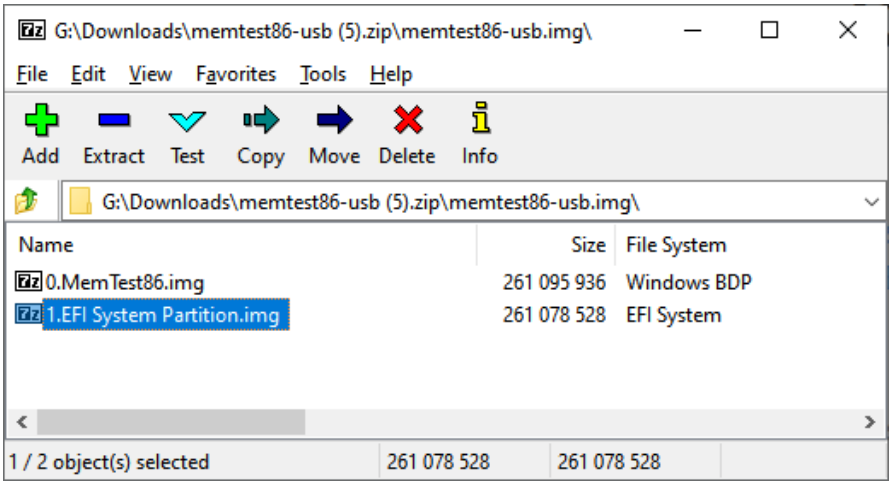


Extract the .iso file and copy it to your Ventoy Partition 1 - e.g. \ISO\MEMTEST folder.

3. Download the latest 'Free' version...

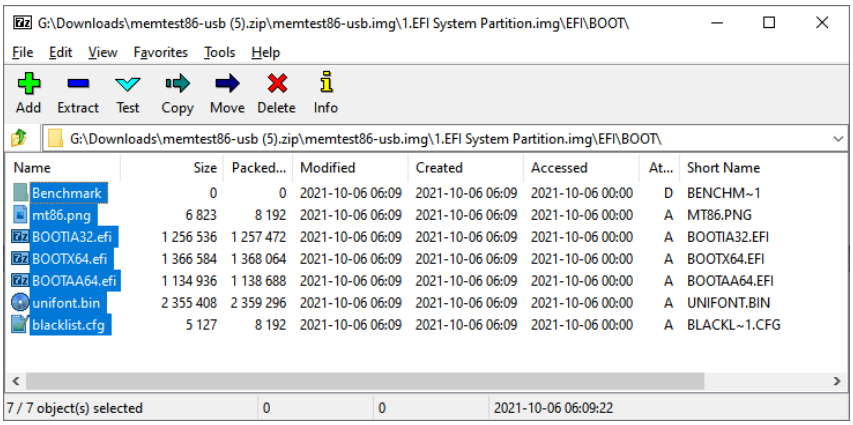


Use 7-zip to extract the EFI files from the download:

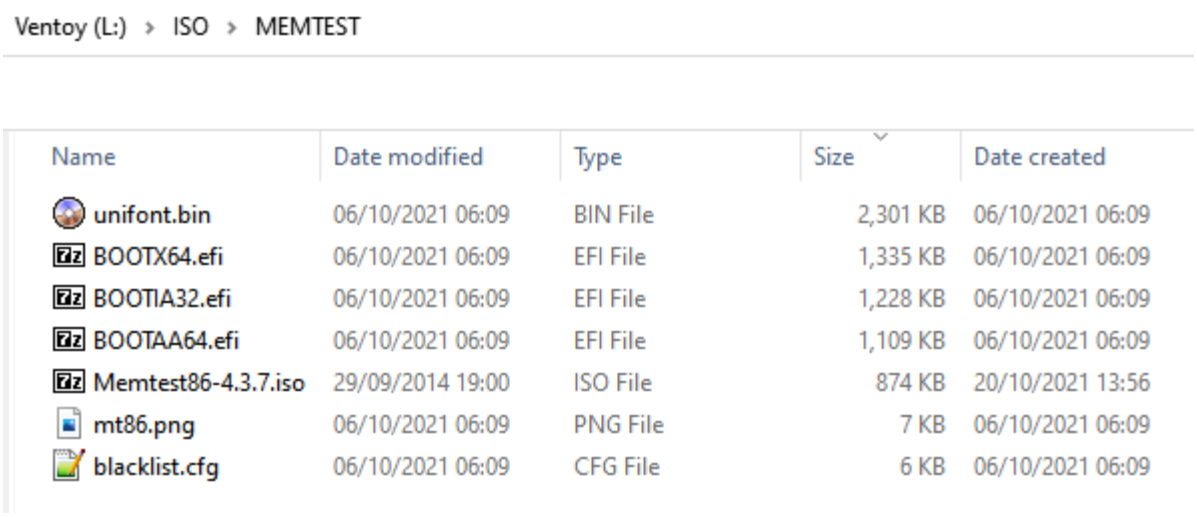


*Note: Ventoy can also boot from some .img files. In this case you could simply extract the '**memtest86-usb.img**' file from the ZIP file and copy it to your Ventoy drive - however in this exercise we will see how to boot from EFI files...*

Now copy all files from the EFI folder inside the zip file to a new folder on your Ventoy drive.



I made a \ISO\MEMTEST folder:



Note that the BOOTAA64.efi file is for UEFI64 ARM platforms, so you can delete it if you have no ARM systems (Ventoy does support ARM64 UEFI too).

Each EFI file contains the entire MemTest86 code, but the font, .png and blacklist.cfg files are also used by MemTest86, if they are found in the same folder as the EFI file. The blacklist.cfg contains a list of 'bad' BIOSes which don't play well with MemTest86.

The ISO and EFI files can be renamed (just don't change their file extensions).

For instance:

MemTest86 v4.3.7 (Legacy only) < Memtest86-4.3.7.iso renamed
MemTest86 v9.3 (UEFI64) < BOOTX64.EFI renamed
MemTest86 v9.3 (UEFI32) < BOOTIA32.EFI renamed

You can also blacklist the legacy ISO file for UEFI mode boots by using the Ventoy.json file (see later chapter).

Files with a .EFI extension are not displayed by Ventoy if you have booted in Legacy mode.

Chapter 7 - Hide some files/folders from Ventoy

If you have just a few payload files on your Ventoy USB drive, then the normal 'List Mode' view will be satisfactory. However, if you have much more than ten ISO files, you may find it is easier to use 'Tree View Mode' by pressing F3 in Ventoy.

If you have used the suggested \ISO folder structure, when you press F3 you will see a menu similar to this...

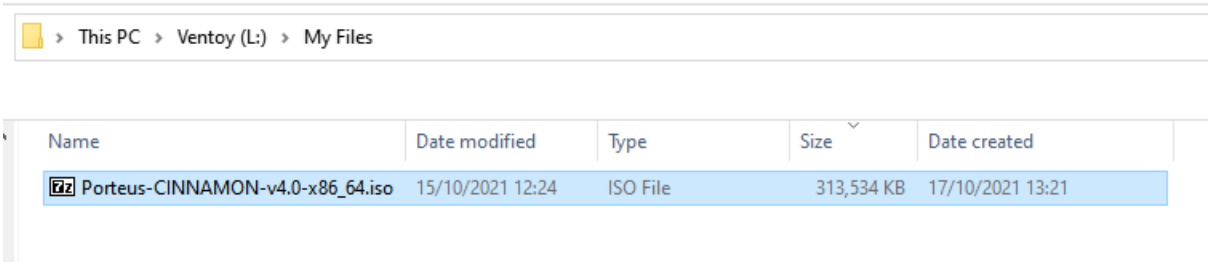
If you select the 'DIR [ISO]' menu entry, you should see...

and then you can select one of those folders to view the ISO files inside that folder.

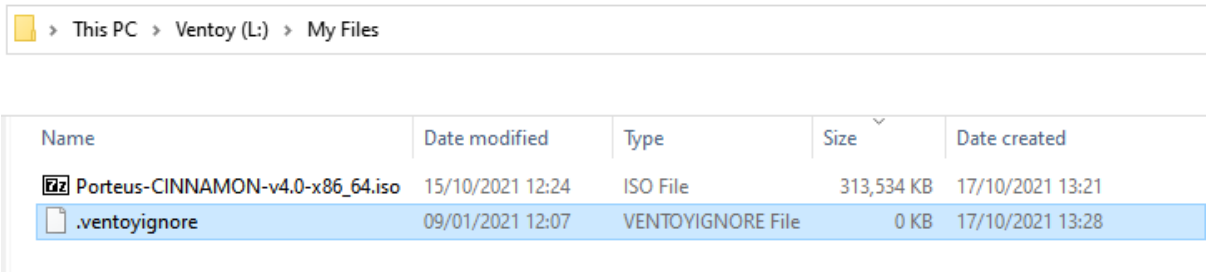
Ventoy will only list 'valid' files (non-corrupted ISO files) which have valid file extensions (such as .iso) but, by default, it will enumerate all files and all folders in the entire partition. If you have a large USB drive containing lots of files and folders, this may slow down the appearance of the initial Ventoy menu quite a lot!

Also, you may have payload files in some folders which you do not want Ventoy to list in the menu system.

Suppose we have a folder in the root called \My Files and in this folder we have more files and folders and older .iso files, etc.



To prevent Ventoy from enumerating all the files and folders under \My Files, we can add a dummy file named **.ventoyignore** (see below).



If you create this file (which can be an empty file) and copy it to the \My Files folder, then any file or folders within the folder '\My Files' will not be listed in the Ventoy menu and it may also greatly speed up the performance of the Ventoy menu system if you have lots of files and folders under that folder.

You should copy the .ventoyignore file to all your other 'non-payload' folders that you have on your Ventoy USB drive (you only need to copy it to the top-level folders).

Chapter 8 - The Ventoy user configuration file (ventoy.json)

The main user configuration file for Ventoy must be created by the user and it must be located at:

\ventoy\ventoy.json on Partition 1.

The Ventoy website has more documentation on this file here.

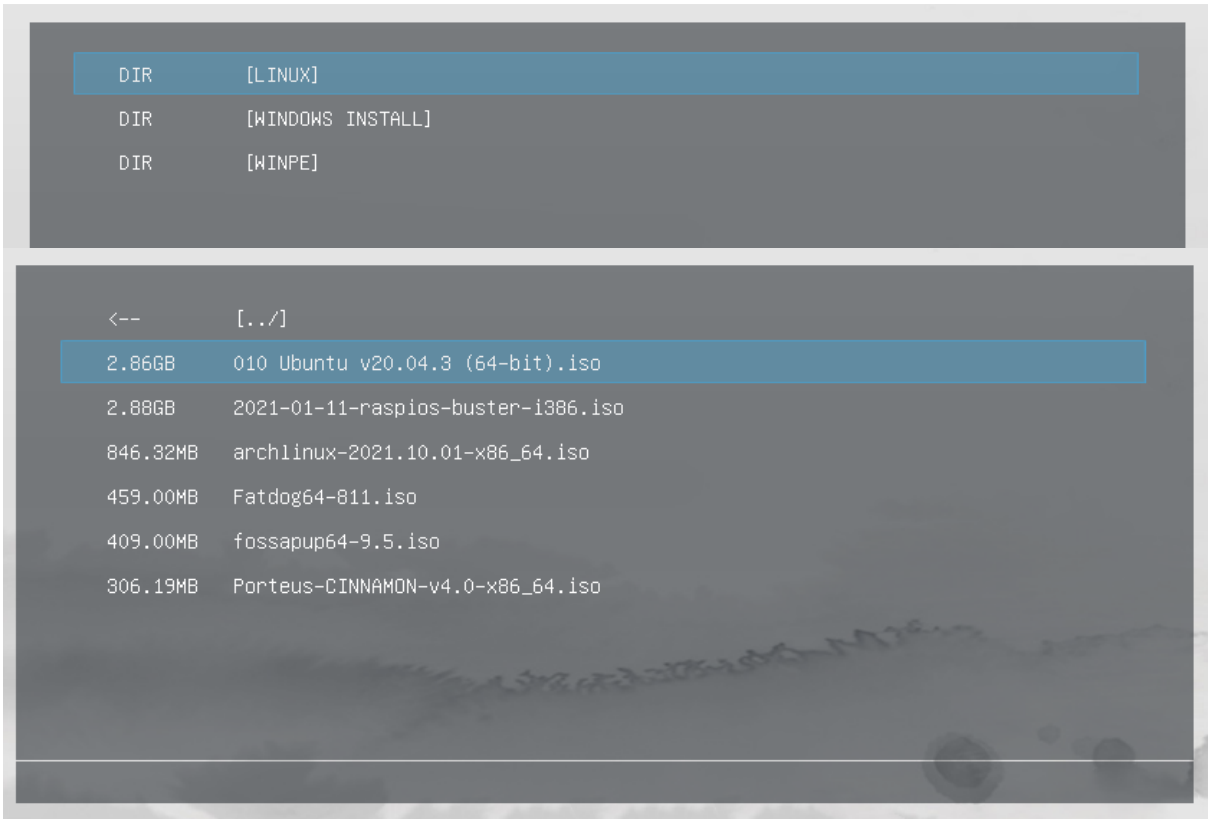
The path (\ventoy) and filename (ventoy.json) is (now) case sensitive, and the text inside the .json file is also case sensitive, so be careful when you specify keys, folder and filenames within a .json file.

Here is a small example which sets the default view mode to Tree View and does not display files if they begin with ._ (period underscore).

Also only the files under the \ISO folder will be enumerated and a UK keyboard layout will be expected in the example below:

```
{
    "control": [
      { "VTOY_DEFAULT_MENU_MODE": "1"},
      { "VTOY_FILT_DOT_UNDERSCORE_FILE": "1"},
      { "VTOY_TREE_VIEW_MENU_STYLE": "0" },
      { "VTOY_DEFAULT_SEARCH_ROOT": "/ISO" },
      { "VTOY_DEFAULT_KBD_LAYOUT": "QWERTY_UK" }
    ]
}
```

Note that ISO files in the root (top level) of the USB drive will now not be listed. Only files\folders starting at \ISO will be shown in the menu system...

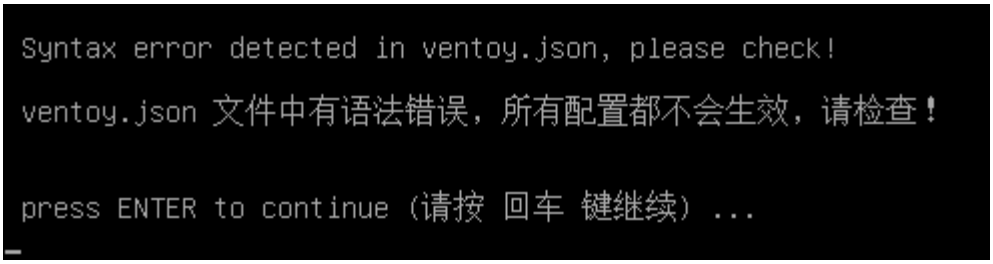


If you do not want the file size listed, then set VTOY_TREE_VIEW_MENU_STYLE to 1.

Ventoy.json syntax rules:

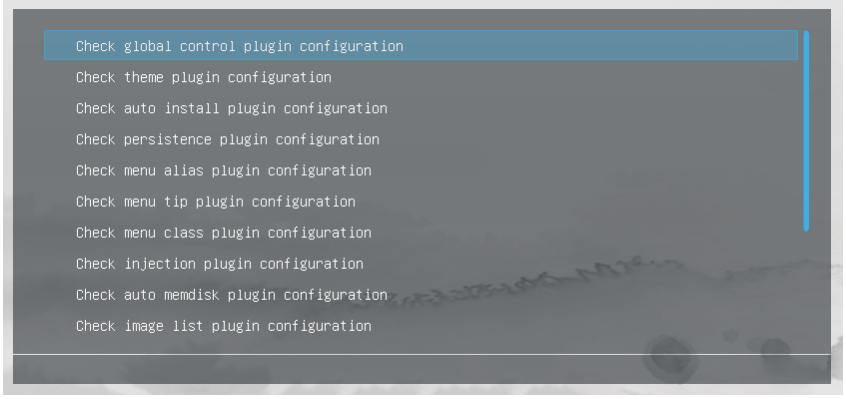
- Paths use a forward slash /, same as Linux (not backslash as used by Windows).
- Whitespace (Space, Horizontal tab, Line feed or New line or Carriage return) anywhere does not matter (it could all be on one line if you like!).
- Data names and value pairs should be in double-quotes.
- The file must start with { and end with }.
- Each *object* begins with { and ends with }.
- Each keyword *name* is enclosed in double-quotes and is followed by a colon (:) and a *value* in double-quotes ("fred" : "1").
- An *array* begins with [and ends with].
- If there is more than one *"name":"value"* pair, each line must end with a comma (,) EXCEPT FOR THE LAST ENTRY.
- Entries are case sensitive. e.g. an incorrect entry of { "VTOY_DEFAULT_MENU_mode": "1" }, will be ignored by Ventoy but the menu will not be shown in Tree View.

The syntax of the ventoy.json file will be checked by Ventoy when it first loads it and it will warn you if it detects a syntax error...



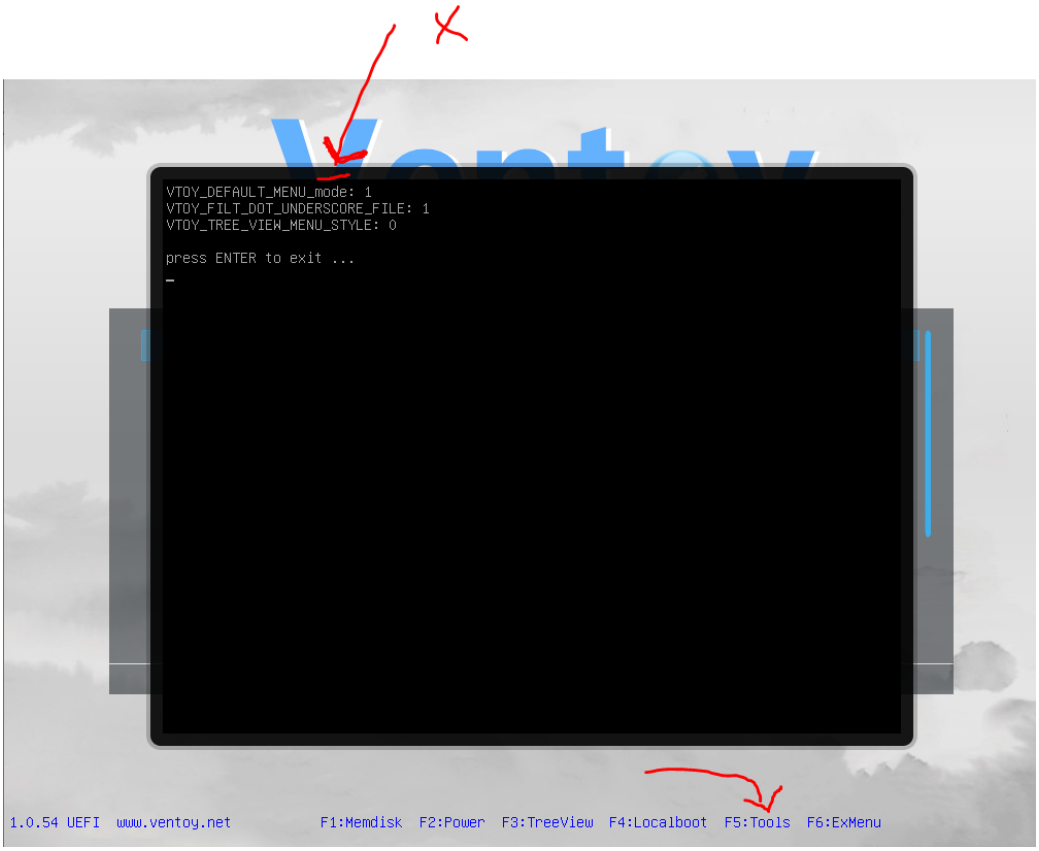
However, it will only detect simple syntax errors. It does not detect invalid case or misspelling. You can check the individual entries using the **F5 Tools menu** to narrow down whereabouts the error is located...

Each section inside the Ventoy.json file can then be checked individually...



Here is the "control" section output - which looks correct - no error is reported...

But in the example below, I have used lowercase instead of uppercase, so Ventoy will simply ignore that entry (no error is reported by Ventoy - the syntax is correct but the KEY name is not recognised because it is not all in uppercase)...



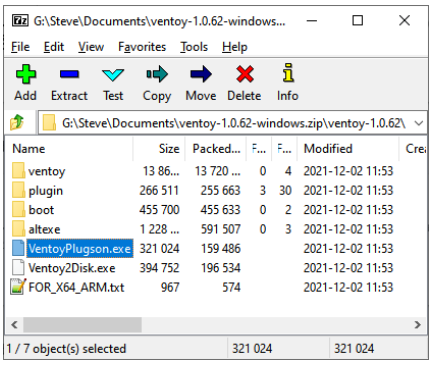
JSON Syntax checker

You can also check your json file using an online syntax checker such as https://www.json.cn (you may need to use this a lot, so bookmark it now!)

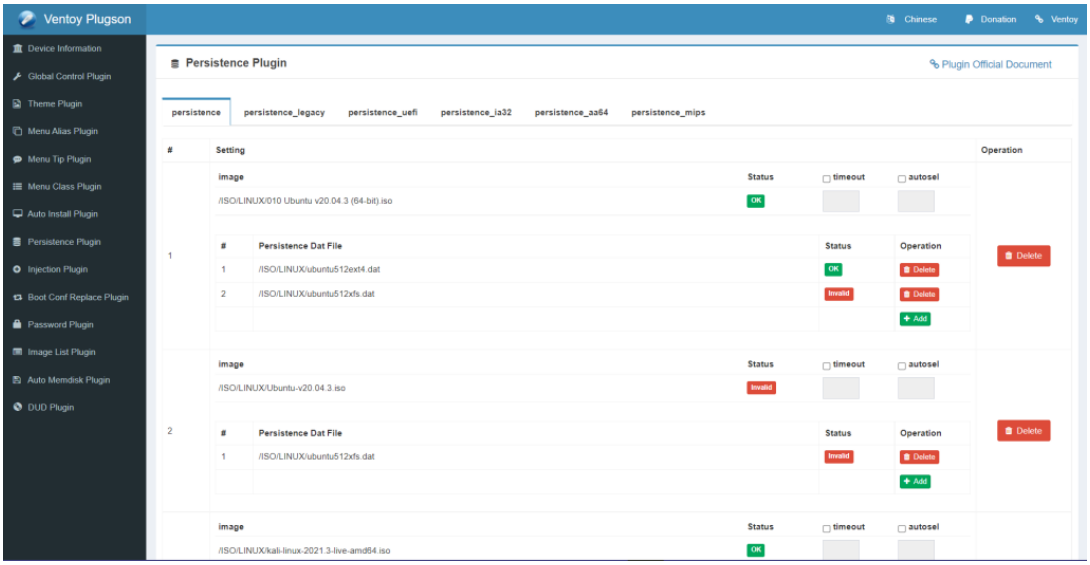
The Pro alternative is here.

VentoyPlugson

Starting with Ventoy 1.0.62 there also is a web-based app called VentoyPlugson which allows you to easily configure the ventoy.json file. You can find the file after extracting all files from the .zip download...



The VentoyPlugson.exe will cause a browser page to be opened. The screenshot below shows a ventoy.json file which contains persistence entries (some of which are marked as 'invalid' because the persistence file does not exist!)...



Note: This is a new app in v1.0.62 and so may contain some bugs/typos, etc.

The following table is taken from the Ventoy Global control plugins website page (check website for latest version)...

Key	Type	Description
VTOY_DEFAULT_MENU_MODE	STRING	Default menu display mode, "0":ListView Mode "1":TreeView Mode
VTOY_TREE_VIEW_MENU_STYLE	STRING	Menu style in TreeView mode. "0":with DIR and file size prefix "1":No DIR and file size Default is "0"
VTOY_FILT_DOT_UNDERSCORE_FILE	STRING	Filter for files with prefix ._ in name. This will be useful when you use macOS (a lot of ._xxx file generated when you copy files) "0":Don't filter "1":Filter
VTOY_SORT_CASE_SENSITIVE	STRING	Case sensitive when sort the ISO files or directories. Default is "0" (case insensitive) "0":case insensitive "1":case sensitive
VTOY_MAX_SEARCH_LEVEL	STRING	Max subdirectory level when search for image files. It's value can be: max 0 1 2 3 ... default is : max By default, Ventoy will search all the directories and sub directories recursively no matter how deep the directory level is. You can use this parameter to set a max-depth for the search path. max : Maximum, search all the directories and subdirectories. This is Ventoy's default value. 0 : Only search files in the root and don't search any subdirectories. 1 : Search up to level 1 of subdirectories. 2 : Search up to level 2 of subdirectories. 3 : Search up to level 3 of subdirectories. ... If VTOY_DEFAULT_SEARCH_ROOT is set at the same time then the level is counted from VTOY_DEFAULT_SEARCH_ROOT .
VTOY_DEFAULT_SEARCH_ROOT	STRING	The root path where to search the iso files. By default, Ventoy will search all the directories and subdirectories in the USB. This will be very slow when you have huge number of files in the USB. In this case, you can put all the iso files in one directory and use this to specify the search path. Ventoy will only search this directory and its subdirectories for iso files.
VTOY_MENU_TIMEOUT	STRING	Menu timeout (seconds). By default no timeout is set. When you set it to 10 for example, the first image will be selected an booted after 10 seconds.
VTOY_DEFAULT_IMAGE	STRING	Default selected image path. Normally used with VTOY_MENU_TIMEOUT. It can be ISO/WIM/VHD/IMG ... and supported in both ListView mode and TreeView mode. Attention, when VTOY_DEFAULT_SEARCH_ROOT is set, VTOY_DEFAULT_IMAGE must be in the directory (or sub-directory) of VTOY_DEFAULT_SEARCH_ROOT, otherwise it will not take effect. Besides, you can use some special values as descripted in following Special VTOY_DEFAULT_IMAGE.
VTOY_VHD_NO_WARNING	STRING	"0": Show a warning message if the partition is not NTFS when booting VHD(x) file. "1": No warning message default is "0"
VTOY_DEFAULT_KBD_LAYOUT	STRING	Keyboard Layout, default is QWERTY_USA supported layouts: QWERTY_USA AZERTY CZECH_QWERTY CZECH_QWERTZ DANISH DVORAK_USA FRENCH GERMAN ITALIANO JAPAN_106 LATIN_USA PORTU_BRAZIL QWERTY_UK QWERTZ QWERTZ_HUN QWERTZ_SLOV_CROAT SPANISH SWEDISH TURKISH_Q VIETNAMESE

VTOY_FILE_FLT_ISO	STRING	"0": List .iso files "1": Filter .iso files. Default is "0"
VTOY_FILE_FLT_WIM	STRING	"0": List .wim files "1": Filter .wim files. Default is "0"
VTOY_FILE_FLT_EFI	STRING	"0": List .efi files "1": Filter .efi files. Default is "0"
VTOY_FILE_FLT_IMG	STRING	"0": List .img files "1": Filter .img files. Default is "0"
VTOY_FILE_FLT_VHD	STRING	"0": List .vhd(x) files "1": Filter .vhd(x) files. Default is "0"
VTOY_FILE_FLT_VTOY	STRING	"0": List .vtoy files "1": Filter .vtoy files. Default is "0"
VTOY_WIN11_BYPASS_CHECK	STRING	"0": Do not bypass Windows 11 hardware check. "1": Bypass Windows 11 hardware check. The default value is "0" When set to 1, Ventoy will create the following registries to bypass Windows 11 hardware check when install. Only available for standard Windows 11 ISO file. HKEY_LOCAL_MACHINE\SYSTEM\Setup\LabConfig\BypassRAMCheck HKEY_LOCAL_MACHINE\SYSTEM\Setup\LabConfig\BypassTPMCheck HKEY_LOCAL_MACHINE\SYSTEM\Setup\LabConfig\BypassSecureBootCheck HKEY_LOCAL_MACHINE\SYSTEM\Setup\LabConfig\BypassCPUCheck HKEY_LOCAL_MACHINE\SYSTEM\Setup\LabConfig\BypassStorageCheck
VTOY_HELP_TXT_LANGUAGE	STRING	The language of the help text when press Ctrl+h. Default is "en_US", refer Ventoy Help Text for details.
VTOY_LINUX_REMOUNT	STRING	"0": I don't need to access the image partition after boot. "1": I need to access the image partition after boot. Default is "0". This option is only available for Linux distro image files. By default, the image partition where the ISO files locate can not be accessed after boot. When you try to mount it you will get device busy error. This is due to linux kernel restriction (device-mapper module). If you select 1 here, Ventoy will try to bypass the restriction with some special mechanism. But it should be noted that, this is an experimental feature and is not fully tested.

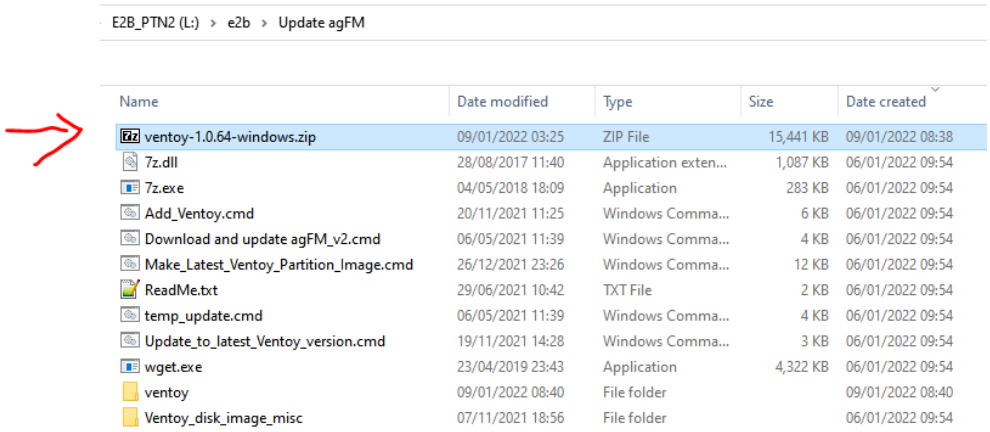
Note that you can specify a default image to boot to and a default timeout.

Using Plugson with an Easy2Boot USB drive

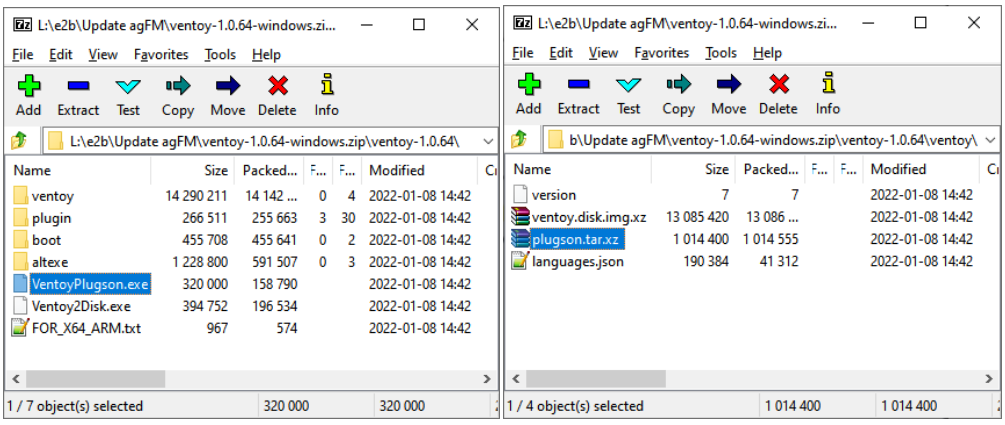
PLEASE NOTE: The Plugson application does not recognise the E2B USB drive unless you first convert it to an 'official' Ventoy drive.

The instructions below assume you have a Windows system (Windows 10 is needed if you have an E2B USB *Flash drive*).

1. Ensure you have agFM installed on your E2B USB drive and run **\e2b\Update agFM\ Download and update agFM_v2.cmd** on the second partition to get the latest version of agFM.
2. Run the script on the second partition:
   \e2b\Update agFM\Make_Latest_Ventoy_Partition_Image.cmd
   to make the three Ventoy .img files and also add the grubfm_multiarch.iso file when prompted.
3. If \ventoy\Ventoy_Plugson.exe already exists on Partition 1 then skip to Step 4.
   You should now find the latest Ventoy zip file is in the \e2b\Update agFM folder.

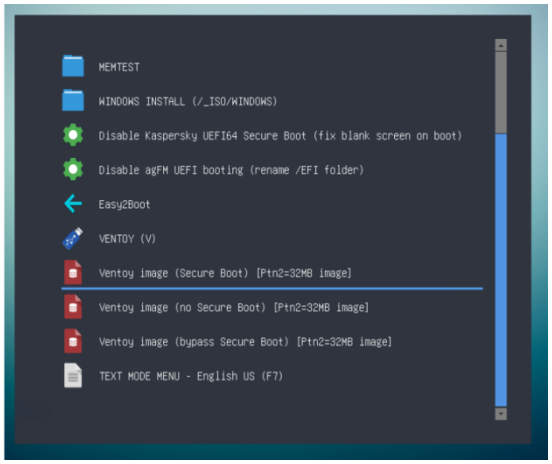


You need to extract the following files from the zip file and copy them to the E2B **Partition 1** volume:
ZIP \Ventoy-1.0.64\VentoyPlugson.exe -> E2B:**\ventoy\VentoyPlugson.exe**
ZIP \Ventoy-1.0.64\ventoy\plugson.tar.xz -> E2B:**\ventoy\ventoy\ plugson.tar.xz**



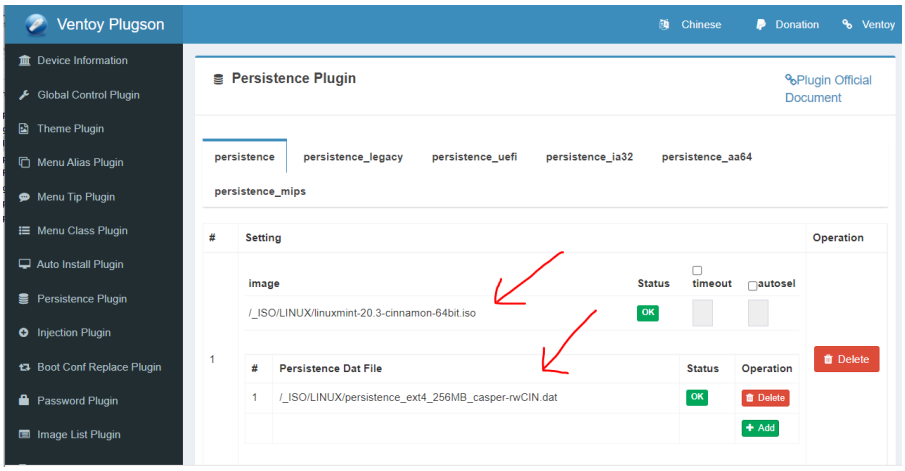
4. Boot from the E2B USB drive and load the agFM menu system (you can use a real system, VMWare or VirtualBox+VMUB).

5. Within the agFM menu system, select one of the three Ventoy images files and allow the system to reboot.



6. Now connect the E2B USB drive to a Windows system and double-click on the **\ventoy\VentoyPlugson.exe** file on *Partition 1* to launch the browser-based application.

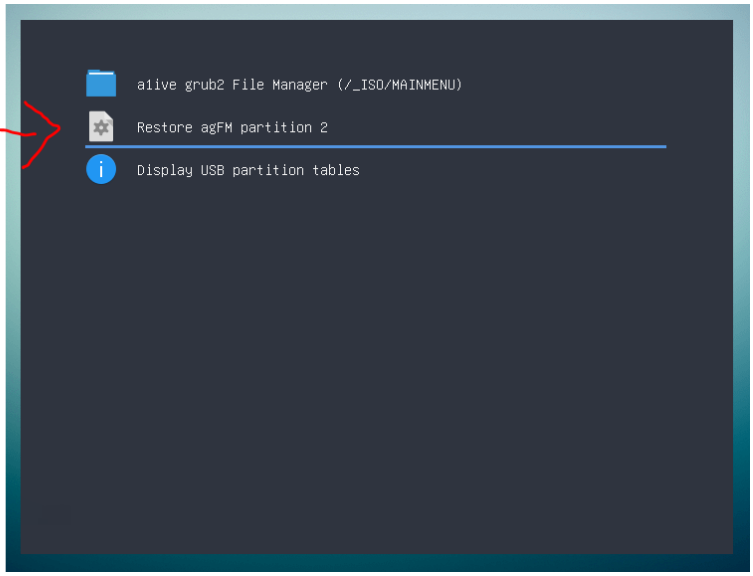
7. VentoyPlugson should detect the E2B USB drive now and any changes you make in the browser will be written to the \ventoy\ventoy.json file immediately.



8. You can test Ventoy by booting from the USB drive now to check your configuration changes. The USB drive will now directly Legacy or UEFI-boot straight to Ventoy as it has now been converted to a Ventoy USB drive!

9. To restore the USB drive back to an 'Easy2Boot' drive with the correct Partition 2 and agFM, boot to Ventoy and run **### Restore agFM Partition 2 using grubfm_multiarch.iso** (see screenshot below).

Press 1 and ENTER to restore Partition 2. Then choose the menu entry below to restore the E2B partition 2:



10. You can now run 'Ventoy for Easy2Boot' from the E2B USB drive and the ventoy.json configuration file will still take effect.

Note: 'Ventoy for Easy2Boot' may contain an older version of Ventoy and so some newer Ventoy configuration options may not work if they are not supported in the older version used by 'Ventoy for Easy2Boot'.

Multimode

Since Ventoy can Legacy boot, UEFI64 boot and boot on some other non-Intel platforms as well, you may wish to specify different default settings for different platforms. This can be done using the multimode options.

For instance, you can ask Ventoy to load a different theme for different platforms:

Take the theme plugin for example, you can set a theme object in the ventoy.json and it will take effect in all BIOS modes.

```
{
    "theme": {
        "file": "/ventoy/theme/theme.txt",
        "gfxmode": "1920x1080"
    }
}
```

Also, you can set theme_legacy, theme_uefi, theme_ia32, theme_aa64 and theme_mips which take effect in x86 Legacy BIOS, x86_64 UEFI, IA32 UEFI, ARM64 UEFI and MIPS64 UEFI mode.

```
{
    "theme_legacy": {
        "display_mode": "CLI"
    },
    "theme_uefi": {
        "file": "/ventoy/theme/themeEFI64.txt",
        "gfxmode": "1920x1080"
    },
    "theme_ia32": {
        "file": "/ventoy/theme/themeEFI32.txt",
        "gfxmode": "1920x1080"
    },
    "theme_aa64": {
        "file": "/ventoy/theme/themeARM64.txt",
        "gfxmode": "1920x1080"
    },
    "theme_mips": {
        "file": "/ventoy/theme/themeMIPS.txt",
        "gfxmode": "1920x1080"
    }
}
```

In the example above, a legacy boot will use the menu in text mode (for best compatibility and fastest response).

I encourage you to study the various plugin pages on the Ventoy website to see what other configuration options are possible. Some (but not all) will be covered later on in this eBook...

Chapter 9 - Ubuntu with persistence

Usually, when you boot from a Live CD or Live USB key, all modifications are discarded when you reboot.

The persistence file (or persistence partition) allows you to keep your preferences, changes, installed apps and data even after a reboot. A persistence file will be 'mounted' by Linux and the file will contain a file system onto which Linux can read and write files as if it were a real disk.

Any changes that you make to the Linux OS - for example, saving a file to your desktop, changing the settings in an application, or installing a program - will be stored in the persistence overlay file. Whenever you boot the USB drive on any computer, your files, settings and installed programs will still be there.

If you are just using a USB drive to install Linux and then running it from your hard drive afterwards or want to run a Live version without it remembering any changes then you don't need persistence.

Persistence is an ideal feature if you want to keep a live Linux system on a USB drive and then use it on different PCs. You won't have to set up the Desktop environment (display, language, keyboard, network, etc.) each time you boot from the ISO.

There are a few limitations. You can't modify system files, like the kernel. You can't perform major system upgrades which may update the kernel or boot code (if you try, it may make your USB drive unbootable!). You also can't install hardware drivers. However, you can install most applications. You can even update most installed applications, so you can be sure your persistent USB drive has the latest version of your preferred web browser, for instance. You also need to be careful when shutting down or rebooting the 'LiveCD' Linux - because file writes are cached by Linux (delayed writes), the persistence file may not have been written to with all changes immediately - so you must never just switch off your PC or unplug the Ventoy USB drive without first clicking on 'Shutdown' within Linux and making sure that it has properly closed all open files.

Tip: If you have problems shutting down or rebooting Linux in a controlled manner, use the '*sudo sync*' command to flush the write buffers to all drives before disconnecting the USB drive or pressing the power button.

Note that *if the persistence file becomes corrupted, you will not be able to boot the ISO* until you either replace or create a new persistence file or you do not boot the ISO with the persistence file.

The persistence file may need to be a special name or contain a special filesystem with a special volume label and special contents (depending on the Linux distro used). Ubuntu-based distros expect a persistence file to contain a file system with a volume label of *casper-rw*. Fedora expects a label of *vtoycow* when used with Ventoy and you may have to edit the grub2 menu to add an extra kernel parameter (*selinux=0*). Some linux distros such as CloneZilla and Kali expect the persistence file to contain an already formatted file system and a special persistence.conf file within it.

Usually, 300 MB is enough to install some software and keep your changes but you can use several GBs is you wish. If you intend to use Linux+Persistence a lot, then 'bigger is better' because it is difficult to 'compress' it or purge any old, redundant changes from the persistence file.

Exercise 10: Ubuntu 64-bit ISO with persistence

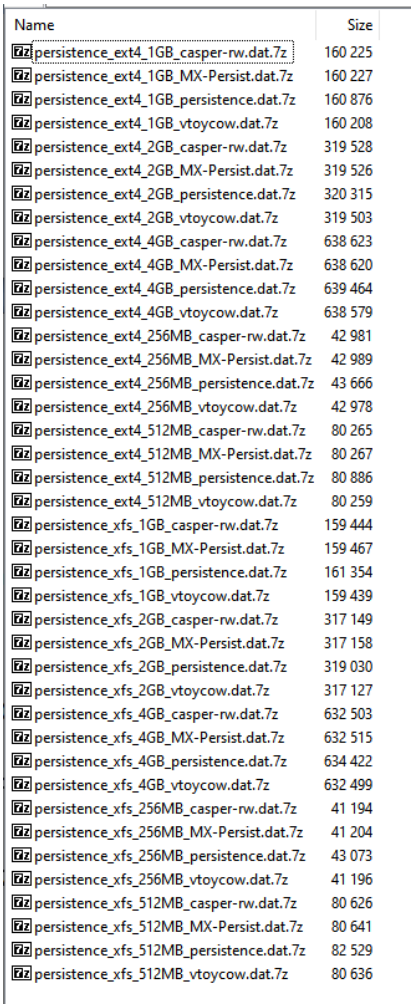
1. I will assume that you have already placed a suitable Ubuntu ISO on your Ventoy USB drive and have checked that it boots correctly as a simple .ISO file. In this example, I have used ubuntu-20.04.3-desktop-amd64.iso and placed it in the \ISO\LINUX folder.

2. We now need to place a persistence file on the USB drive. The Ventoy website provides a table of different Linux distros and the volume labels that they will look for.

In that table, Ubuntu-based ISOs are shown to require a pre-formatted filesystem inside the persistence file and which have a 'casper-rw' volume label.

Ventoy provides a zip file containing ready-made persistence files of different types and sizes for use with different distros.

Download the images.zip file from here and then click on it to view the files inside using 7-Zip...



Name	Size
persistence_ext4_1GB_casper-rw.dat.7z	160 225
persistence_ext4_1GB_MX-Persist.dat.7z	160 227
persistence_ext4_1GB_persistence.dat.7z	160 876
persistence_ext4_1GB_vtoycow.dat.7z	160 208
persistence_ext4_2GB_casper-rw.dat.7z	319 528
persistence_ext4_2GB_MX-Persist.dat.7z	319 526
persistence_ext4_2GB_persistence.dat.7z	320 315
persistence_ext4_2GB_vtoycow.dat.7z	319 503
persistence_ext4_4GB_casper-rw.dat.7z	638 623
persistence_ext4_4GB_MX-Persist.dat.7z	638 620
persistence_ext4_4GB_persistence.dat.7z	639 464
persistence_ext4_4GB_vtoycow.dat.7z	638 579
persistence_ext4_256MB_casper-rw.dat.7z	42 981
persistence_ext4_256MB_MX-Persist.dat.7z	42 989
persistence_ext4_256MB_persistence.dat.7z	43 666
persistence_ext4_256MB_vtoycow.dat.7z	42 978
persistence_ext4_512MB_casper-rw.dat.7z	80 265
persistence_ext4_512MB_MX-Persist.dat.7z	80 267
persistence_ext4_512MB_persistence.dat.7z	80 886
persistence_ext4_512MB_vtoycow.dat.7z	80 259
persistence_xfs_1GB_casper-rw.dat.7z	159 444
persistence_xfs_1GB_MX-Persist.dat.7z	159 467
persistence_xfs_1GB_persistence.dat.7z	161 354
persistence_xfs_1GB_vtoycow.dat.7z	159 439
persistence_xfs_2GB_casper-rw.dat.7z	317 149
persistence_xfs_2GB_MX-Persist.dat.7z	317 158
persistence_xfs_2GB_persistence.dat.7z	319 030
persistence_xfs_2GB_vtoycow.dat.7z	317 127
persistence_xfs_4GB_casper-rw.dat.7z	632 503
persistence_xfs_4GB_MX-Persist.dat.7z	632 515
persistence_xfs_4GB_persistence.dat.7z	634 422
persistence_xfs_4GB_vtoycow.dat.7z	632 499
persistence_xfs_256MB_casper-rw.dat.7z	41 194
persistence_xfs_256MB_MX-Persist.dat.7z	41 204
persistence_xfs_256MB_persistence.dat.7z	43 073
persistence_xfs_256MB_vtoycow.dat.7z	41 196
persistence_xfs_512MB_casper-rw.dat.7z	80 626
persistence_xfs_512MB_MX-Persist.dat.7z	80 641
persistence_xfs_512MB_persistence.dat.7z	82 529
persistence_xfs_512MB_vtoycow.dat.7z	80 636

You can see that for each size there are different types:

- casper-rw.dat - for Ubuntu-based, Linux Mint, Elementary OS, Zorin, Kaspersky Rescue disk, etc.
- MX-Persist.dat - for MX Linux
- persistence.dat - for Kali and CloneZilla (contains persistence.conf file)
- vtoycow.dat - for Arch Linux and Fedora

If you have a distro that is not mentioned, you will need to check if the LiveCD ISO supports persistence and what type of persistence file is expected. Also you should check what extra kernel parameters may be required.

The Ventoy images.zip file contains examples with both ext4 and xfs file systems inside. You can use either ext4 or xfs for Ubuntu as long as it is one of the 'casper-rw' ones.

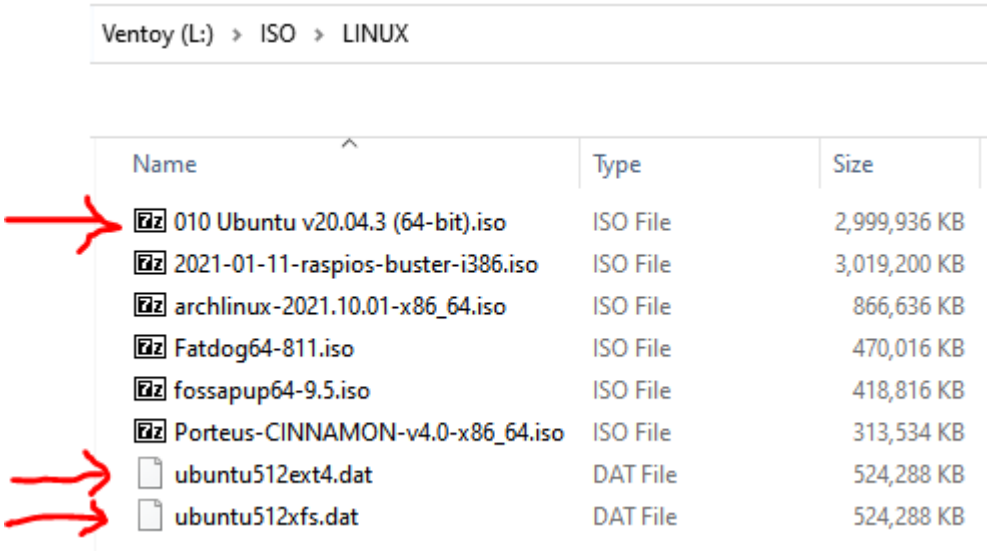
I extracted both the ext4 and ext4 casper-rw .dat files and placed them in the same folder as the Ubuntu ISO file (just to check that both would work!). Note that the final file must end in .**dat** (not .zip) and you can give them any filename you like.

You may prefer to place all your .dat persistence files in a separate folder (e.g. \persistence). You could also add the .ventoyignore file to that folder so that it was not enumerated by the Ventoy menu system for faster boot times.

When using a persistence file with LiveCD ISOs and USB drives, it is essential that the file system will recover well if there is a sudden power loss or if the USB drive is unexpectedly removed from the USB port. I extracted both these files and renamed them:

ubuntu512ext4.dat
ubuntu512xfs.dat

Here is my \ISO\LINUX folder with the two new, empty .dat files:



Note that I have used 512MB .dat files and renamed them. You can use any name for the files that you like.

For instance, you may want one persistence file for 'work' and one for 'games', or one for 'Jim' and one for 'Mary'.

You can just use one persistence file if you wish.

3. Now edit the \ventoy\ventoy.json file to include the name of the ISO and the persistence .dat file(s). I have made entries two different Ubuntu ISO files and two persistence files. One of the persistence files is used by both Ubuntu ISOs:

```
{
	"control": [{
			"VTOY_DEFAULT_MENU_MODE": "1"
		},
		{
			"VTOY_FILT_DOT_UNDERSCORE_FILE": "1"
		},
		{
			"VTOY_TREE_VIEW_MENU_STYLE": "0"
		},
		{
			"VTOY_DEFAULT_SEARCH_ROOT": "/ISO"
		},
		{
			"VTOY_DEFAULT_KBD_LAYOUT": "QWERTY_UK"
		}
	],
	"persistence": [
	{
		"image": "/ISO/LINUX/010 Ubuntu v20.04.3 (64-bit).iso",
		"backend": [
		"/ISO/LINUX/ubuntu512ext4.dat",
		"/ISO/LINUX/ubuntu512xfs.dat"
		]
	},
	{
		"image": "/ISO/LINUX/Ubuntu-v20.04.2.iso",
		"backend": "/ISO/LINUX/ubuntu512xfs.dat"
	}
	]

}
```

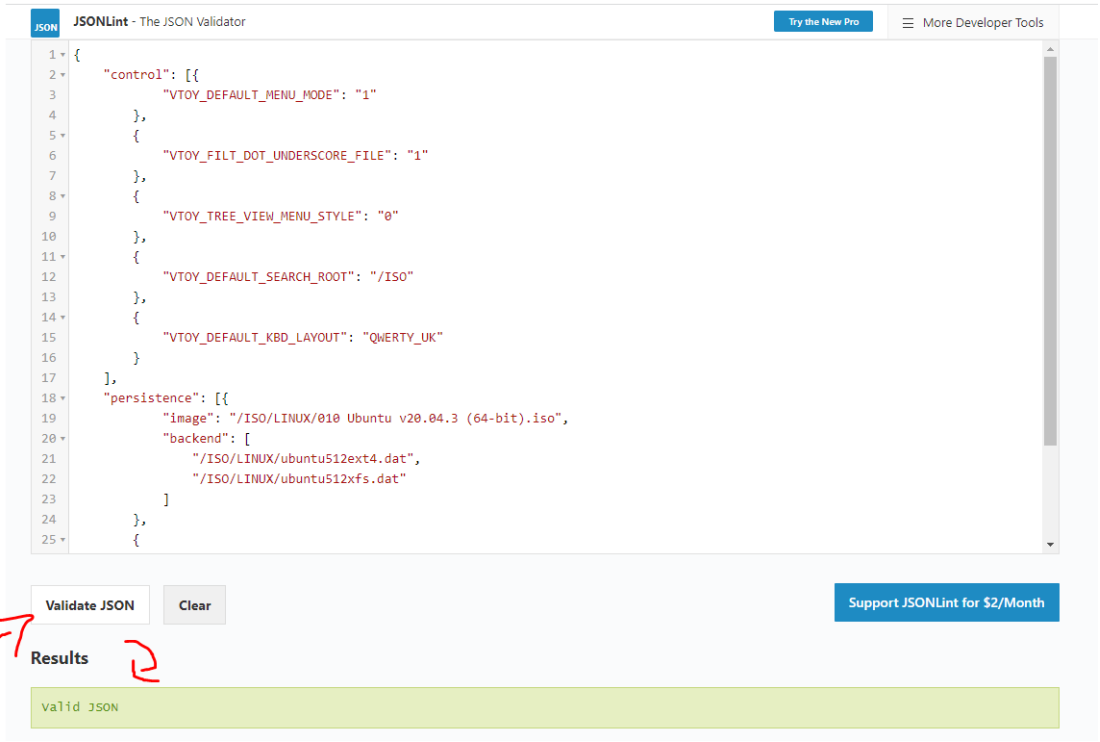
Note: If you have only one persistence file, remove the line in red.

You can download a complete **example ventoy.json file** used with this eBook here (delete any unwanted lines).

If the ISO file /ISO/LINUX/Ubuntu-v20.04.2.iso does not exist, then Ventoy will simply ignore the second entry.

If you use the same persistence file for two different versions of the Ubuntu ISO, then the ISOs need to be extremely similar or strange things may happen...

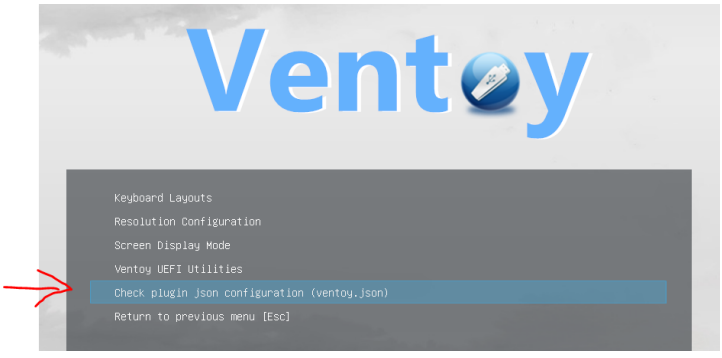
Personally, I find that the Json syntax is *extremely* difficult to work with, so I have to always check it using an online syntax checker (e.g. at https://jsonlint.com/) because I usually get it wrong! Here is an example of this online checker with just one ISO listed in the persistence section and I have two persistence files to choose from ('valid JSON' in green = passed)...



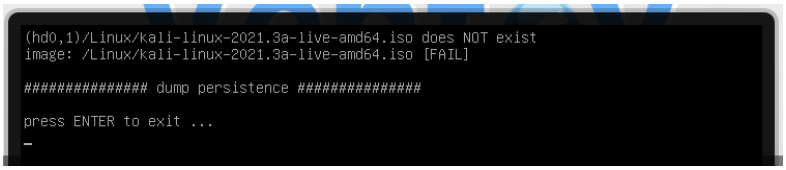
Tip: This 'Pro' version is an alternative free version (click on the *'Lint'* button to check the syntax after pasting in the text from the Ventoy.json file).

Note that the "persistence" section requires the data to be surrounded by [ ] square brackets or else it will be ignored by Ventoy and the F5 Tools json persistence check menu will report a 'not array type 4 error' because no array data structure using [ ] was found in that section.

Tip: Ventoy allows you to check each section of the ventoy.json file using the F5 Tools - Check plugin json configuration menu:

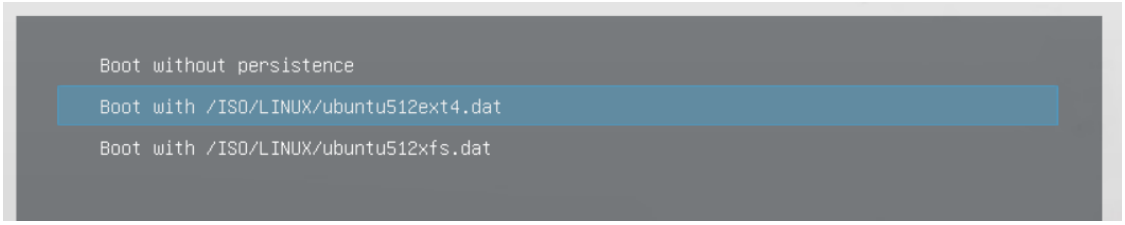


If there is a problem, you will see a message, e.g.



4. Now Legacy or UEFI64-boot to Ventoy and select the Ubuntu ISO.

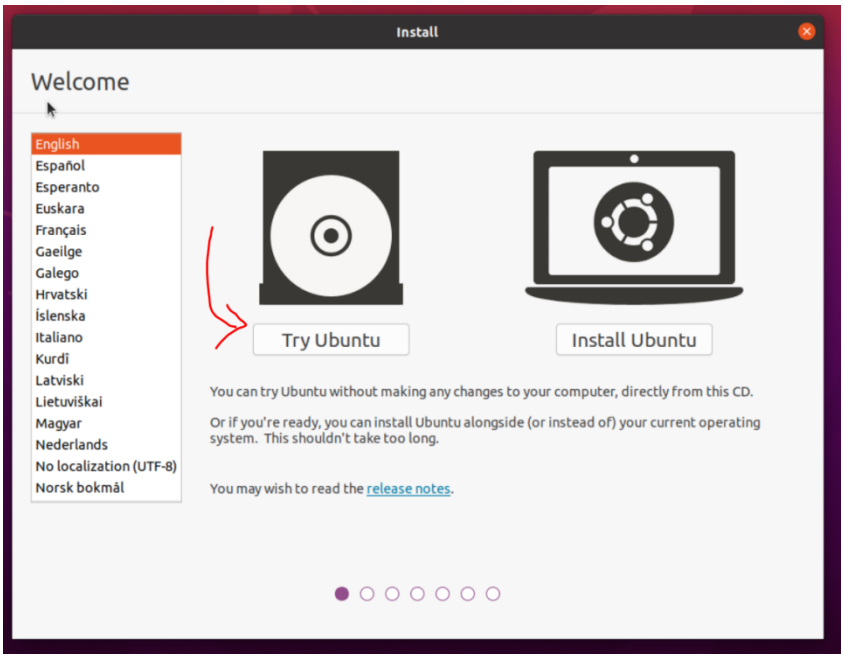
You should then see that Ventoy now gives you a secondary menu with some persistence options:



Choose a .dat file and press ENTER. If you don't see this menu, try the F5 Tools json checker to check for errors in your ventoy.json file!

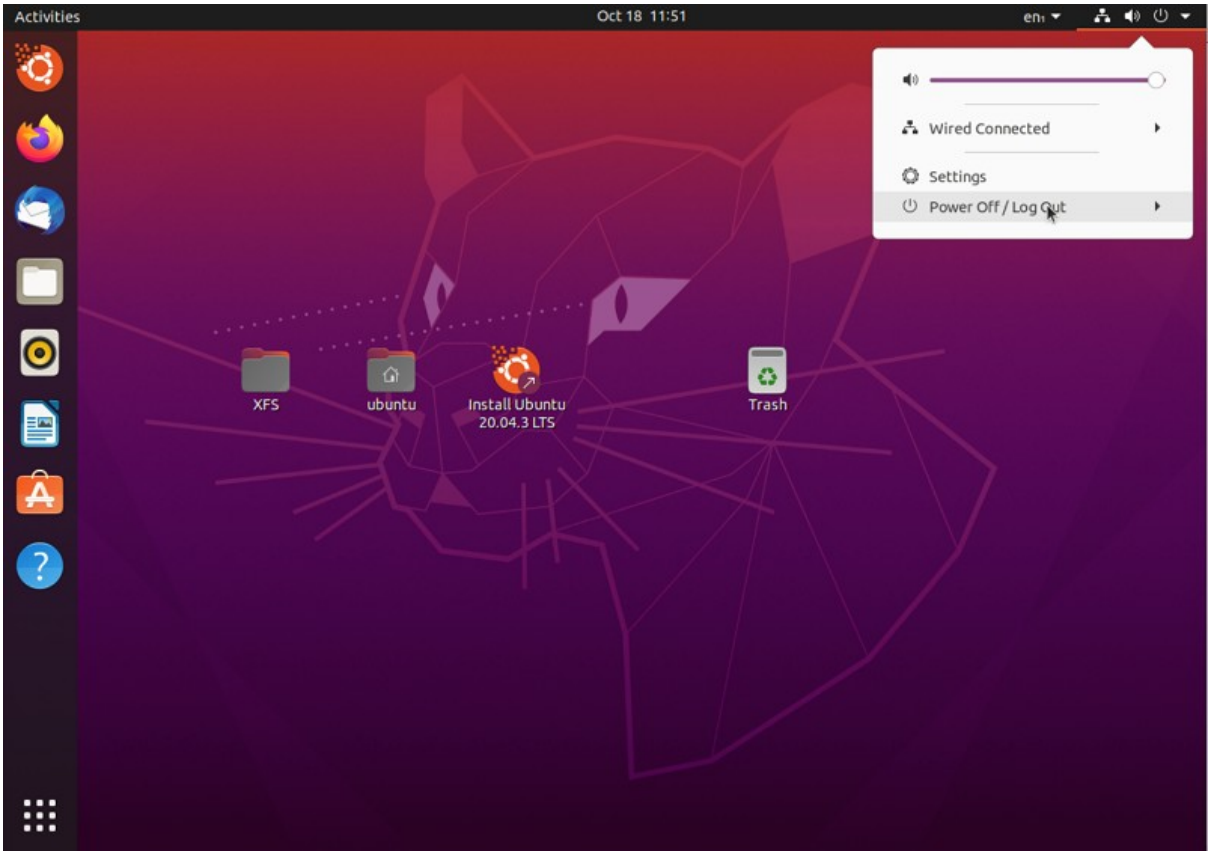
Note: Use a real system for testing (not a VM). If the VM is working in snapshot mode then you may see an error on boot or persistence may not work. VBox v5 + VMUB works OK however.

5. When Ubuntu boots, click on the 'Try Ubuntu' button...



To test that persistence is working, re-arrange the Desktop icons and then right-click somewhere on the Desktop to create a new folder ('New Folder').

Then use the Shutdown button (top-right) to Power Off - Restart.



When you reboot, you should see your new folder is still there (e.g. folder XFS in screenshot above) and that the new Desktop icon positions have been remembered.

You can also install other software using the Orange 'A' icon on the far left of the screen (e.g. Chromium browser) if you have internet access.

6. Finally, after you have set up Ubuntu as you like it, with your apps and WiFi settings, etc. and have shutdown Ubuntu, I strongly recommend that you *make a copy of the persistence .dat file* that you have used and save it to a safe place and onto a different disk (not the USB drive). In the event of the persistence file becoming corrupt or if you lose the USB drive, you can simply restore the persistence file and will not have lost hours of work!

Note: The same persistence file will work for both Legacy booting and UEFI64 booting.

Although you can configure Ventoy so that the persistence menu will be automatically selected, I do not recommend this because if your persistence file becomes corrupted, you will not be able to boot to the Ubuntu ISO at all (until you replace the .dat file or edit the Ventoy.json file).

Key	Type	Description
`image`	`STRING`	The full path of the iso file. This option supports fuzzy matching, please refer to About Path Matching
`backend`	`STRING or ARRAY`	The persistence backend image file path for the iso file. Can be a single string or string arra
`autosel`	`INTEGER`	Optional. If you set it, that means you auto select the corresponding option in the prompt menu and the prompt menu will NOT be shown. 0: boot without persistence backend image file 1: boot with the 1st persistence backend image file 2: boot with the 2nd persistence backend image file

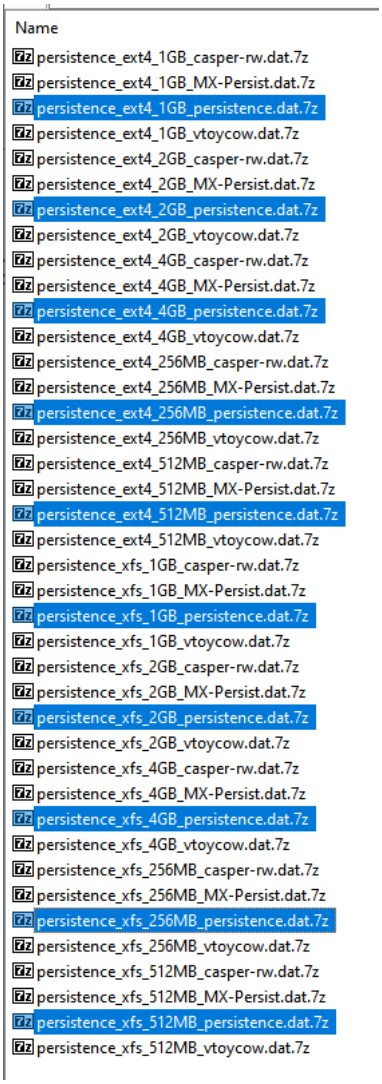
Exercise 11: Kali 64-bit ISO with persistence

Kali Linux is quite a large ISO (4GB) with many integrated utilities. It is mainly used for penetration testing (i.e. 'hacking').

1. Download the Kali Live CD ISO and copy the ISO file to the \ISO\LINUX folder of your USB drive, e.g.

\ISO\LINUX\kali-linux-2021.3-live-amd64.iso

2. Extract one of the persistence_*_persistence.dat files from the images.zip file (see previous Exercise). I have highlighted the suitable ones in blue below - just pick one...



and extract the .dat file from the .7z file.

3. Copy the *persistence.dat file to the _ISO\LINUX folder and give it a nicer name.

4. Modify your \ventoy\ventoy.json file as required (see below):

```
{
	"control": [{
			"VTOY_DEFAULT_MENU_MODE": "1"
		},
		{
			"VTOY_FILT_DOT_UNDERSCORE_FILE": "1"
		},
		{
			"VTOY_TREE_VIEW_MENU_STYLE": "0"
		},
		{
			"VTOY_DEFAULT_SEARCH_ROOT": "/ISO"
		},
		{
			"VTOY_DEFAULT_KBD_LAYOUT": "QWERTY_UK"
		}
	],
	"persistence": [{
		"image": "/ISO/LINUX/010 Ubuntu v20.04.3 (64-bit).iso",
		"backend": [
			"/ISO/LINUX/ubuntu512ext4.dat",
			"/ISO/LINUX/ubuntu512xfs.dat"
		]
	}, {
		"image": "/ISO/LINUX/Ubuntu-v20.04.3.iso",
		"backend": "/ISO/LINUX/ubuntu512xfs.dat"
	}, {
		"image": "/ISO/LINUX/kali-linux-2021.3-live-amd64.iso",
		"backend": "/ISO/LINUX/kali.dat"
	}]
}
```

Again, note that we *must* use [and] to enclose the data fields following the "persistence" keyword.

5. Now boot to Ventoy, select the Kali ISO and the kali.dat file and then choose the 'Live USB Persistence' option from the Kali boot menu:

Tip: Use a real system if you have problems booting under a Virtual Machine.

Note: Ventoy does not support *SecureBoot* with Kali + Persistence, so if you UEFI-boot you must *disable SecureBoot* in the BIOS settings first.

Check persistence is working by moving one of the Desktop icons and rebooting.

Check both UEFI and Legacy booting work. Kali may have problems booting on some computers, in which case try booting *without* persistence and use the Advanced options menu with different settings.

Chapter 10 - Windows Installs using an XML file

Ventoy allows you to automate the start up of some ISOs by using a script file.

Distros	Template	Example	Notes
Windows	Unattend XML	unattended.xml	Windows
RHEL8/CentOS8	Kickstart script	kickstart8.cfg	Distros based on them are also supported.
RHEL7/CentOS7/Fedora	Kickstart script	kickstart7.cfg	Distros based on them are also supported.
RHEL6/CentOS6	Kickstart script	kickstart6.cfg	Distros based on them are also supported.
Debian/Ubuntu Server	Preseed script	preseed.cfg	Distros based on them are also supported.
SUSE	autoYast XML	autoYast.xml	SLES and openSUSE

An XML file can be useful when booting to a Windows ISO and it can contain many settings which Windows Setup will use during installation without needing to ask the user.

For instance, when you boot to a Windows 8.1 ISO, it will usually ask you for the Product Key and you cannot proceed until you have entered a valid key. However, if you use an XML file which contains a suitable product key, then it will skip that prompt.

Another typical case is when you want install (say) Windows 10 Professional on a computer that originally came from the manufacturer with Windows 8 or 10 Home pre-installed by the OEM. Because the computer will contain the Window Home OEM Product Key embedded into the Non-Volatile RAM of the motherboard, Windows will only allow you to re-install Windows Home. However, if you use an XML file which contains a general purpose Windows Professional product key, *you can force Windows Setup to install Windows Professional instead of Windows Home.*

Windows XML files can be configured to *automatically* wipe and partition the target disk and install windows *without any user interaction* at all.

You can specify many other options in the XML file such as language, country, user account name, password, admin account details, configuration options, etc.

An easy way to make a semi-automated XML file is to use the WAFG website here - just use the appropriate Desktop or Server page for your ISO. Note that if you specify any disk partitions in the XML file, then you will

need one XML file for UEFI installations (which creates GPT partitions) and a different XML file for Legacy\MBR settings (which creates Legacy BIOS compatible MBR partitions).

Exercise 12: Install Windows 10 Professional onto any computer

In the exercise, we are going to force the Windows 10 ISO to always pick and install Windows Professional by using an XML file (even if the computer is licensed for Windows Home).

1. Create a text file with an .xml file extension using Notepad or a similar text editor.

Cut and paste the contents below into the .xml file and save the file to the same folder that contains your Windows 10 ISO file (note that some of the 26 lines wrap over, so always use cut-and-paste).

```
<?xml version="1.0" encoding="utf-8" ?>
<unattend xmlns="urn:schemas-microsoft-com:unattend">

<settings pass="windowsPE">

<component name="Microsoft-Windows-Setup" processorArchitecture="x86"
publicKeyToken="31bf3856ad364e35" language="neutral" versionScope="nonSxS"
xmlns:wcm="http://schemas.microsoft.com/WMIConfig/2002/State"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<UserData>
<AcceptEula>true</AcceptEula>
<ProductKey>
<Key>VK7JG-NPHTM-C97JM-9MPGT-3V66T</Key>
</ProductKey>
</UserData>
</component>

<component name="Microsoft-Windows-Setup" processorArchitecture="amd64"
publicKeyToken="31bf3856ad364e35" language="neutral" versionScope="nonSxS"
xmlns:wcm="http://schemas.microsoft.com/WMIConfig/2002/State"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<UserData>
<AcceptEula>true</AcceptEula>
<ProductKey>
<Key>VK7JG-NPHTM-C97JM-9MPGT-3V66T</Key>
</ProductKey>
</UserData>
</component>

</settings>

</unattend>
```

Notice that this XML file will work for both Legacy and UEFI installs and for 32-bit and 64-bit Windows versions because it contains both component sections: **processorArchitecture="amd64"** and **processorArchitecture="x86"**.

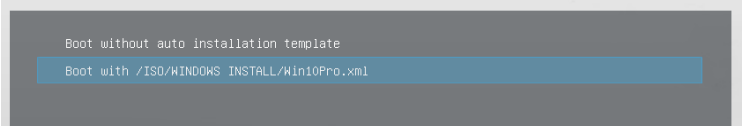
The correct product key to use can be obtained from the WAFG site here. Note however that the keys listed on the site are generic *volume licence* keys and although they will work on Retail/OEM versions of Windows, online activation is slightly more cumbersome (you have to use the 'troubleshooting' activation option in Windows which takes a few more clicks). That is why I have used the generic Microsoft-provided Retail Profession *installation-only* key of VK7JG-NPHTM-C97JM-9MPGT-3V66T in the XML file example above.

2. We now need to edit our \ventoy\ventoy.json file. Here is my sample file:

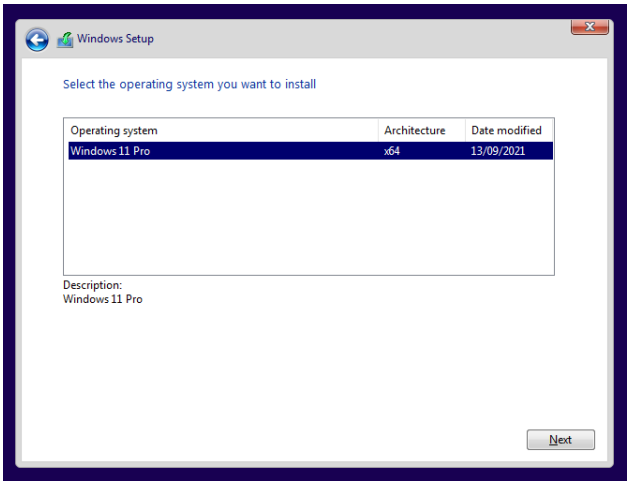
```
{
	"control": [{
			"VTOY_DEFAULT_MENU_MODE": "1"
		},
		{
			"VTOY_FILT_DOT_UNDERSCORE_FILE": "1"
		},
		{
			"VTOY_TREE_VIEW_MENU_STYLE": "0"
		},
		{
			"VTOY_DEFAULT_SEARCH_ROOT": "/ISO"
		},
		{
			"VTOY_DEFAULT_KBD_LAYOUT": "QWERTY_UK"
		}
	],
	"persistence": [{
			"image": "/ISO/LINUX/010 Ubuntu v20.04.3 (64-bit).iso",
			"backend": [
				"/ISO/LINUX/ubuntu512ext4.dat",
				"/ISO/LINUX/ubuntu512xfs.dat"
			]
		},
		{
			"image": "/ISO/LINUX/Ubuntu-v20.04.3.iso",
			"backend": "/ISO/LINUX/ubuntu512xfs.dat"
		}
	],

	"auto_install": [{
		"image": "/ISO/WINDOWS
INSTALL/Win10_EnglishInternational_x64.iso",
		"template": "/ISO/WINDOWS INSTALL/Win10Pro.xml"
	}]
}
```

3. Now boot to Ventoy and select the Window 10 ISO file.

You should now see a secondary menu:



Choose the XML file and press ENTER to boot to Windows Setup. After the Region selection screen, you should see only 'Windows Professional' listed:



You can now proceed to install Windows manually as before.

If you use an XML file made from the WAFG site, you can delete the <DiskConfiguration> and <ImageInstall> sections so that you can specify manually which disk and partition you want to install to. Simply delete all lines between <DiskConfiguration> and </DiskConfiguration> and from <ImageInstall> to </ImageInstall> (including those lines).

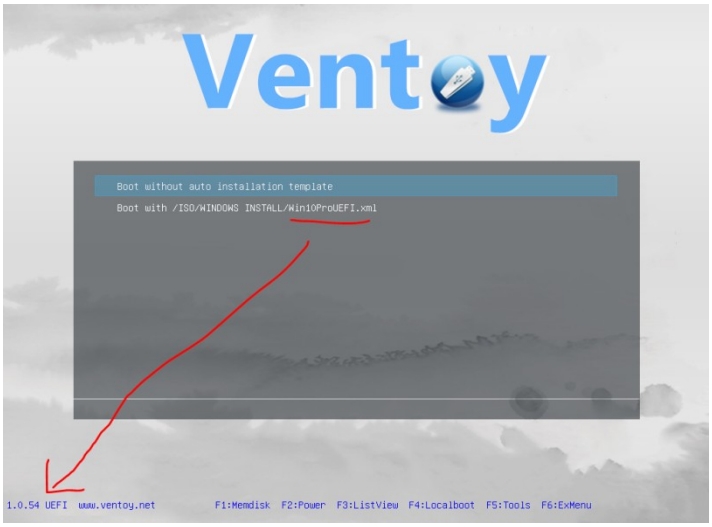
Exercise 13: Platform specific XML file

In the exercise above, it did not matter if we booted via UEFI64 or Legacy, the XML file would still work because it only contained a Product Key, however *if we use an XML file which partitions the target drive*, we will need one XML file for UEFI and a different one for Legacy installs.

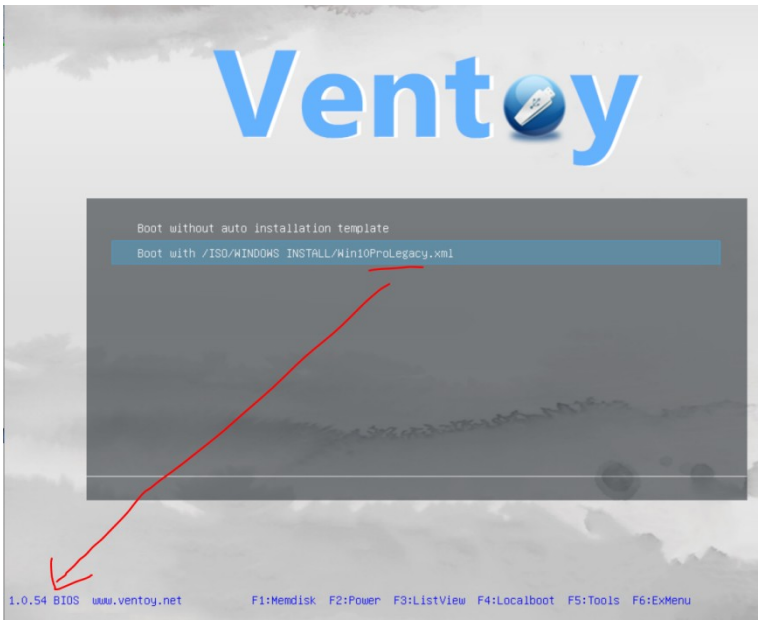
We can modify the Ventoy.json file in this way:

```
{
	"control": [{
			"VTOY_DEFAULT_MENU_MODE": "1"
		},
		{
			"VTOY_FILT_DOT_UNDERSCORE_FILE": "1"
		},
		{
			"VTOY_TREE_VIEW_MENU_STYLE": "0"
		},
		{
			"VTOY_DEFAULT_SEARCH_ROOT": "/ISO"
		},
		{
			"VTOY_DEFAULT_KBD_LAYOUT": "QWERTY_UK"
		}
	],
	"persistence": [{
			"image": "/ISO/LINUX/010 Ubuntu v20.04.3 (64-bit).iso",
			"backend": [
				"/ISO/LINUX/ubuntu512ext4.dat",
				"/ISO/LINUX/ubuntu512xfs.dat"
			]
		},
		{
			"image": "/ISO/LINUX/Ubuntu-v20.04.3.iso",
			"backend": "/ISO/LINUX/ubuntu512xfs.dat"
		}
	],

	"auto_install_legacy": [{
		"image": "/ISO/WINDOWS
INSTALL/Win11_EnglishInternational_x64.iso",
		"template": "/ISO/WINDOWS INSTALL/Win10ProLegacy.xml"
	}],
	"auto_install_uefi": [{
		"image": "/ISO/WINDOWS
INSTALL/Win11_EnglishInternational_x64.iso",
		"template": "/ISO/WINDOWS INSTALL/Win10ProUEFI.xml"
	}]
}
```

Now if we *UEFI boot*, only the UEFI XML file will be offered in the secondary menu.



If we *Legacy boot* to Ventoy and select the ISO file, we should see the Legacy XML file only:



Multiple XML files for each ISO

You can have more than one XML in the JSON file for each ISO, if you wish.

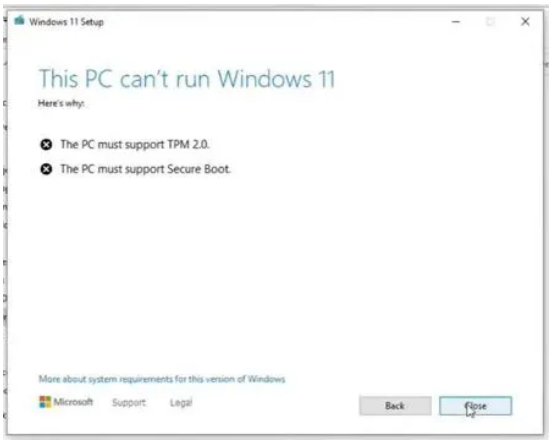
```
{
	"control": [{
			"VTOY_DEFAULT_MENU_MODE": "1"
		},
		{
			"VTOY_FILT_DOT_UNDERSCORE_FILE": "1"
		},
		{
			"VTOY_TREE_VIEW_MENU_STYLE": "0"
		},
		{
			"VTOY_DEFAULT_SEARCH_ROOT": "/ISO"
		},
		{
			"VTOY_DEFAULT_KBD_LAYOUT": "QWERTY_UK"
		}
```

```
	],
	"persistence": [{
		"image": "/ISO/LINUX/010 Ubuntu v20.04.3 (64-bit).iso",
		"backend": [
			"/ISO/LINUX/ubuntu512ext4.dat",
			"/ISO/LINUX/ubuntu512xfs.dat"
		]
	}, {
		"image": "/ISO/LINUX/Ubuntu-v20.04.3.iso",
		"backend": "/ISO/LINUX/ubuntu512xfs.dat"
	}],

	"auto_install_legacy": {
		"image": "/ISO/WINDOWS
INSTALL/Win10_EnglishInternational_x64.iso",
		"template": [
			"/ISO/WINDOWS INSTALL/Win10ProLegacy.xml",
			"/ISO/WINDOWS INSTALL/Win10ProLegacy2.xml"
		]
	},

	"auto_install_uefi": {
		"image": "/ISO/WINDOWS
INSTALL/Win10_EnglishInternational_x64.iso",
		"template": [
			"/ISO/WINDOWS INSTALL/Win10ProUEFI.xml",
			"/ISO/WINDOWS INSTALL/Win10ProUEFI2.xml"
		]
	}
}
```

Ventoy will list the XML files in the secondary menu.

Chapter 11 - Install Windows 11 to systems with no TPM

If you try to run a Windows 11 Install ISO on an old computer, it may display a hardware compatibility error message (perhaps your PC does not have a TPM or Secure Boot or you may not have a UEFI system).



It is possible to tweak Setup so that it will not check for a TPM, Secure Boot or the amount of RAM in your system by adding a registry fragment into WinPE.

This will allow you to install Windows 11 to older systems, however you must be aware that **any Windows 11 Update process that runs after you have fully installed Windows will not work** - if the computer is too old or

does not meet the hardware requirements then Windows 11 will refuse to add any newer hotfixes, drivers and security updates, etc.

The latest versions of Ventoy allow you to automatically add this registry fix by using the VTOY_WIN11_BYPASS_CHECK key in the "control" section of your \ventoy\ventoy.json file:

```
{ "VTOY_WIN11_BYPASS_CHECK": "1" }
```

The parameter, "VTOY_WIN11_BYPASS_CHECK": "1", will install certain WinPE Registry keys that will bypass RAM, TPM, Secure Boot, CPU and Storage checks on the machine.

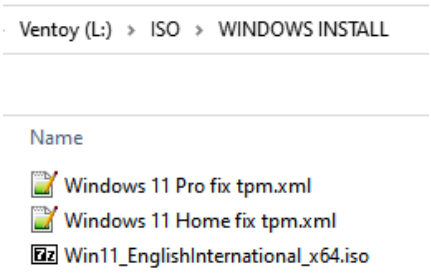
```
HKEY_LOCAL_MACHINE\SYSTEM\Setup\LabConfig\BypassRAMCheck
HKEY_LOCAL_MACHINE\SYSTEM\Setup\LabConfig\BypassTPMCheck
HKEY_LOCAL_MACHINE\SYSTEM\Setup\LabConfig\BypassSecureBootCheck
HKEY_LOCAL_MACHINE\SYSTEM\Setup\LabConfig\BypassCPUCheck
HKEY_LOCAL_MACHINE\SYSTEM\Setup\LabConfig\BypassStorageCheck
```

This next Exercise will also allow you to test Windows 11 on older (unsupported) computers by using an XML file. Using this method you can add any other registry keys or drivers, etc. that you wish.

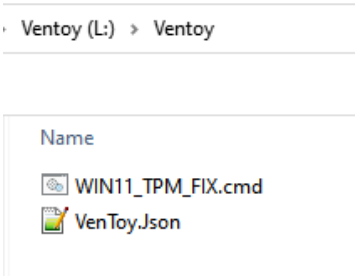
Exercise 14: Install Windows 11 on old hardware

1. Download and copy a Windows 11 ISO file to the \ISO\WINDOWS INSTALL folder.
2. Create XML files containing a RunSynchronousCommand entry and a generic Windows 10/11 install Product Key for Home or Pro
3. You can download the files from here.
4. Place the two XML files in the \ISO\WINDOWS INSTALL folder on the Ventoy Partition 1 volume of the USB drive (same folder as the Win11 ISO).
5. Place the **WIN11_TPM_FIX.cmd** file in the \ventoy folder.
6. Create or edit your \ventoy\ventoy.json text file which contains one or more XML file entries for each Win11 ISO that you want to use, e.g. see the example file on next page.

Check the files are in the correct folders:

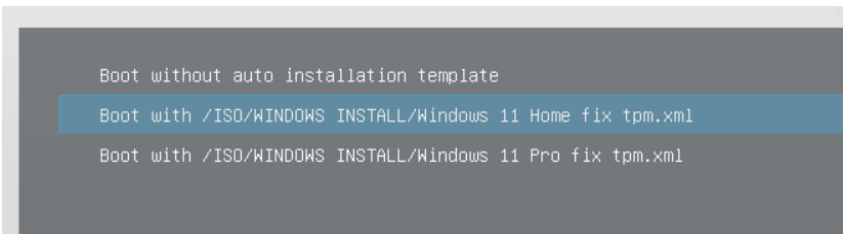


and



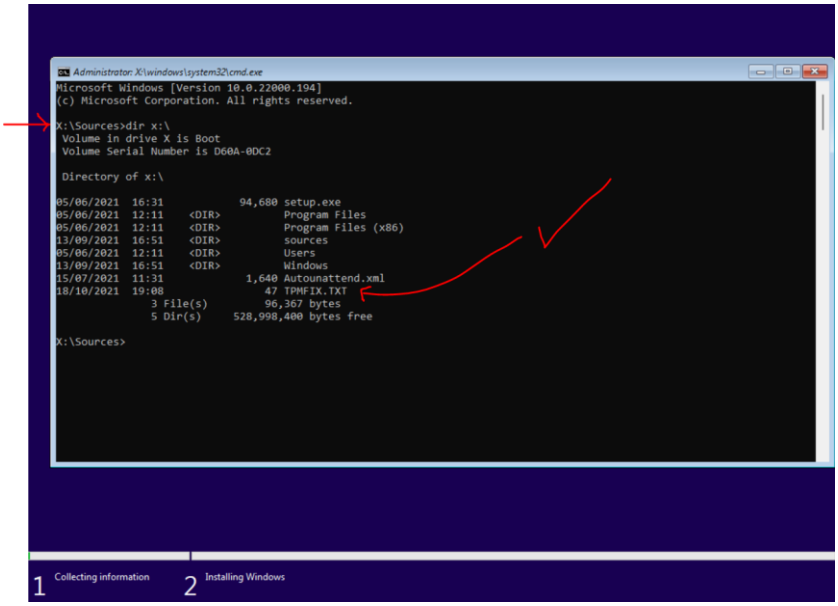
7. Now boot to Ventoy and select the Windows 11 ISO file.

The secondary Ventoy menu should prompt you to select one of the 'tpm' XML files



Ventoy should run the registry fix in the file \ventoy\WIN11_TPM_FIX.cmd and Setup should now install onto a non-TPM system, even in Legacy\MBR mode and with low RAM.

If you want to check that the \ventoy\WIN11_TPM_FIX.cmd file actually ran, then at the Windows Setup GUI screen, press the key chord SHIFT+F10 to open a cmd window and type
dir x:\ (as shown in the screenshot below). The file TPMFIX.TXT will only be present if the registry fix ran.



The Ventoy.json file is shown below:

Ventoy.json

```
{
	"control": [{
			"VTOY_DEFAULT_MENU_MODE": "1"
		},
		{
			"VTOY_FILT_DOT_UNDERSCORE_FILE": "1"
		},
		{
			"VTOY_TREE_VIEW_MENU_STYLE": "0"
		},
		{
			"VTOY_DEFAULT_SEARCH_ROOT": "/ISO"
		},
		{
			"VTOY_DEFAULT_KBD_LAYOUT": "QWERTY_UK"
		}
	],
	"persistence": [{
		"image": "/ISO/LINUX/010 Ubuntu v20.04.3 (64-bit).iso",
		"backend": [
			"/ISO/LINUX/ubuntu512ext4.dat",
			"/ISO/LINUX/ubuntu512xfs.dat"
		]
	}, {
		"image": "/ISO/LINUX/Ubuntu-v20.04.3.iso",
		"backend": "/ISO/LINUX/ubuntu512xfs.dat"
	}],

	"auto_install_legacy": [{
			"image": "/ISO/WINDOWS
INSTALL/Win10_EnglishInternational_x64.iso",
			"template": [
				"/ISO/WINDOWS INSTALL/Win10ProLegacy.xml",
				"/ISO/WINDOWS INSTALL/Win10ProLegacy2.xml"
			]
		},
		{
			"image": "/ISO/WINDOWS
INSTALL/Win11_EnglishInternational_x64.iso",
			"template": [
				"/ISO/WINDOWS INSTALL/Windows 11 Home fix tpm.xml",
				"/ISO/WINDOWS INSTALL/Windows 11 Pro fix tpm.xml"
			]
		}
	],

	"auto_install_uefi": [{
			"image": "/ISO/WINDOWS
INSTALL/Win10_EnglishInternational_x64.iso",
			"template": [
				"/ISO/WINDOWS INSTALL/Win10ProUEFI.xml",
				"/ISO/WINDOWS INSTALL/Win10ProUEFI2.xml"
			]
		},
		{
			"image": "/ISO/WINDOWS
INSTALL/Win11_EnglishInternational_x64.iso",
			"template": [
				"/ISO/WINDOWS INSTALL/Windows 11 Home fix tpm.xml",
```

```
                        "/ISO/WINDOWS INSTALL/Windows 11 Pro fix tpm.xml"
                    ]
                }
        ]

}
```

Chapter 12 - Add WinPE ISOs

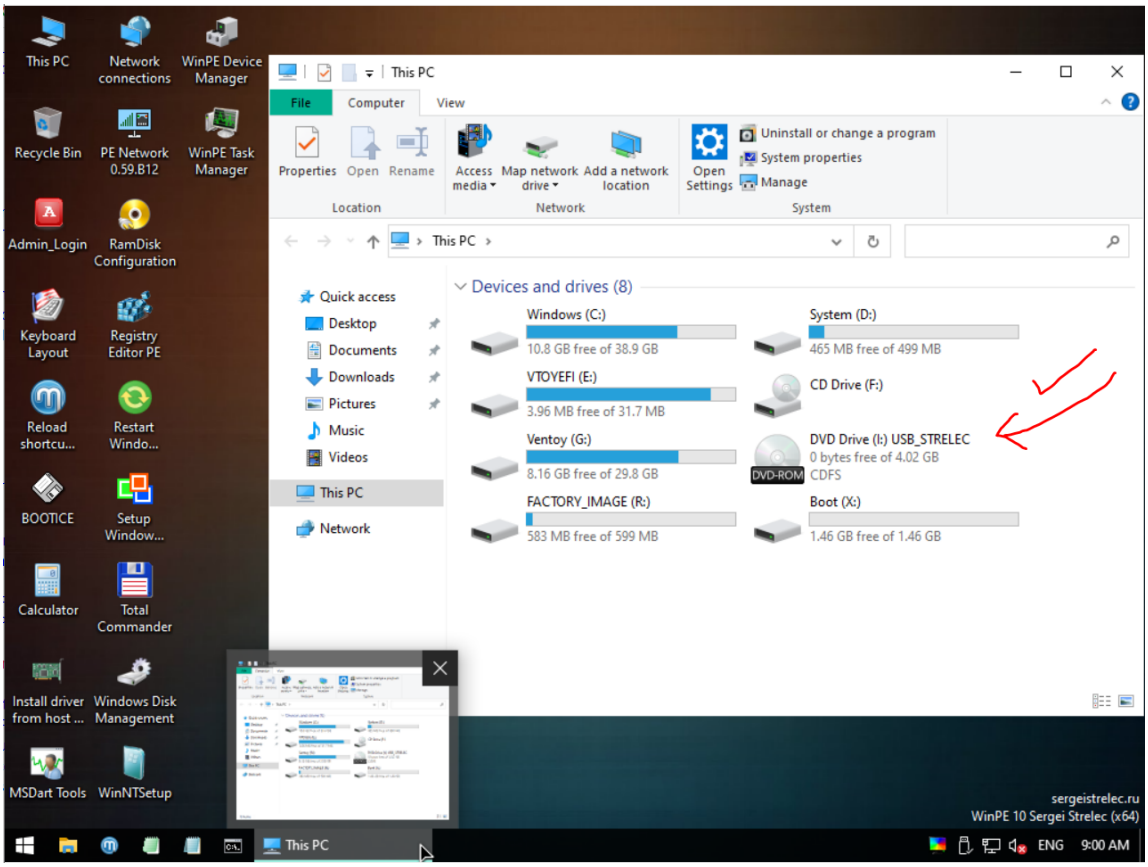
Note: Ventoy seems to have some incompatibilities when UEFI64 booting to WinPE ISOs under an emulator or some Virtual Machines, so always test on a real system if you have problems using a VM.

You can download some WinPE-based ISOs and just copy them to a \ISO\WINPE folder (or wherever you like!).

Strelec WinPE

The Sergei Strelec WinPE ISOs are regularly updated and contain many useful utilities.

Once the Desktop is reached you should see twenty or more icons. If you see less than expected, check that 'This PC' contains a USB_STRELEC virtual DVD Drive (which is actually the mounted ISO file).



Many of the programs and utilities are kept in folders inside the ISO and are not embedded in the boot files. Ventoy should automatically mount the ISO as a virtual DVD so that the programs are available once WinPE has fully booted.

Exercise 15: Add the Strelec WinPE ISO

1. To Download the ISO, first go here and click on the green button...

2. On the new page, scroll down to the section shown below and click on one of the download links...

3. Now just extract and copy the ISO file to any folder on your Ventoy drive and try it.

Hirens Boot CD WinPE

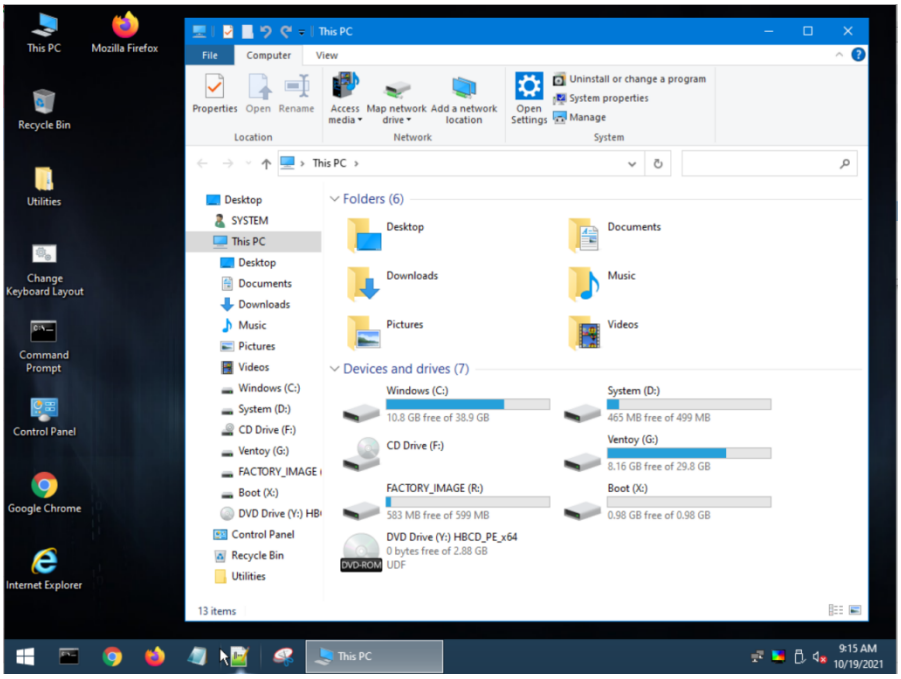
The old Legacy version of HBCD has been updated to a new WinPE-only version which supports UEFI too.

Exercise 16: Add HBCD_PE ISO

1. Download the ISO from here and copy it to the \ISO\WINPE folder.

2. Boot to Ventoy and select the ISO.

3. Once booted to the Desktop, ensure that there is a HBCD_PE_x64 virtual DVD Drive (Y:) in 'This PC'.



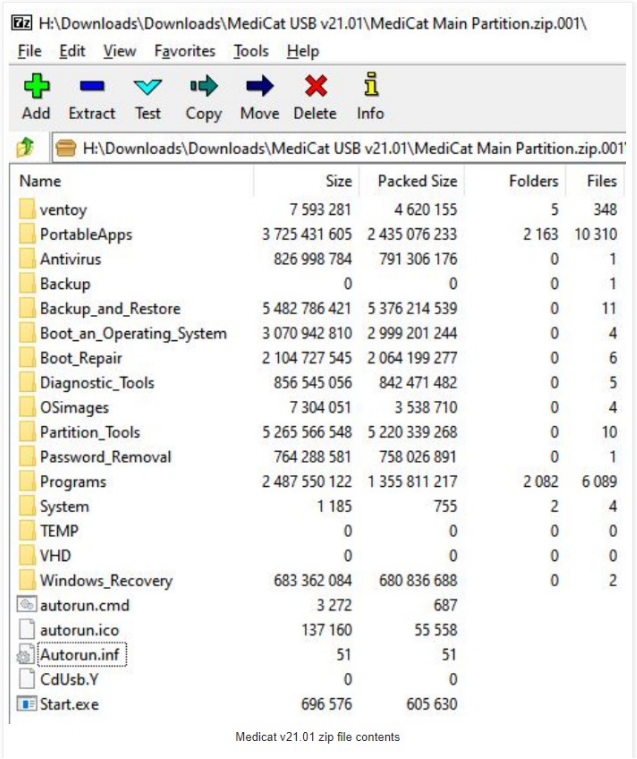
If this is not present then many applications will be missing from the Desktop and Start menu!

Other useful WinPE ISOs

Gandalf RedStone 5 WinPE (download is slow!)

All-in-One System Rescue - for system reapir

MediCat - this now a very large compressed file which is supposed to be written to a USB drive and actually makes a complete, new Ventoy USB drive. You cannot add the image to a ready-made USB drive, but you can use 7Zip to extract the files that you want to add to your Ventoy drive.



Look out for the .wim and .iso files inside the image file and copy them to your Ventoy drive.

Note that the Medicat \ventoy\ventoy.json file contains many entries which you may find interesting to look at and perhaps use in your Ventoy.json file.

There is an Easy2Boot article here which you may also find useful.

Chapter 13 - Add AntiVirus ISOs

AntiVirus ISOs can be copied to the \ISO\ANTIVIRUS folder.

- Kaspersky Rescue Disk - krd**.iso**
- DrWeb - drweb-livedisk-900-cd.iso - Updates are not persistent. However, downloading latest updates is quite quick.
- Eset - eset_sysrescue_live - Reasonably quick to download latest updates.
- Norton Bootable Recovery Tool - **nbrt.iso.** It always downloads the latest updates every time which takes few minutes.
- GData - GDATA_BootCD - Use the 'BootMedium' antivirus ISO
- Comodo Rescue Disk – **MBR-boot only** (no UEFI)

Chapter 14 - Add Chrome OS ISOs (CloudReady and FydeOS)

ChromeOS ISOs are difficult to boot because they are usually in the form of a multiple-partition hybrid ISO which you are supposed to write to a USB drive.

There are a few distributions however, which Ventoy supports:

CloudReady (UEFI64 only)

1. Download a version of CloudReady. I used the free Home version 64-bit Image file from here.

3. Unzip the .ZIP file that you just downloaded and copy the 6GB file to your Ventoy USB drive.

4. Rename the .bin file extension to .**img**.

CloudReady only supports UEFI, so you now need to UEFI64-boot.

Now you should be able to sign into your Google account. If you have two-factor authentication enabled, you will need to confirm the login via your mobile phone (enable Bluetooth and Location detection on your phone).

Once you have logged in successfully, any future boot to the CloudReady img file will just require your Google account email address and password to be entered.

Note: You will need to test using a real system as Virtual Box did not seem to work for me. I successfully tested this on a Lenovo IdeaPad 300.

FydeOS

I used 'mirror 2' to download the 2GB .img.xz file and then unzipped it to make a 7.3GB .img file.

I had problems using the trackpad on the IdeaPad300 with this distro and it appears to be a version intended mainly for China.

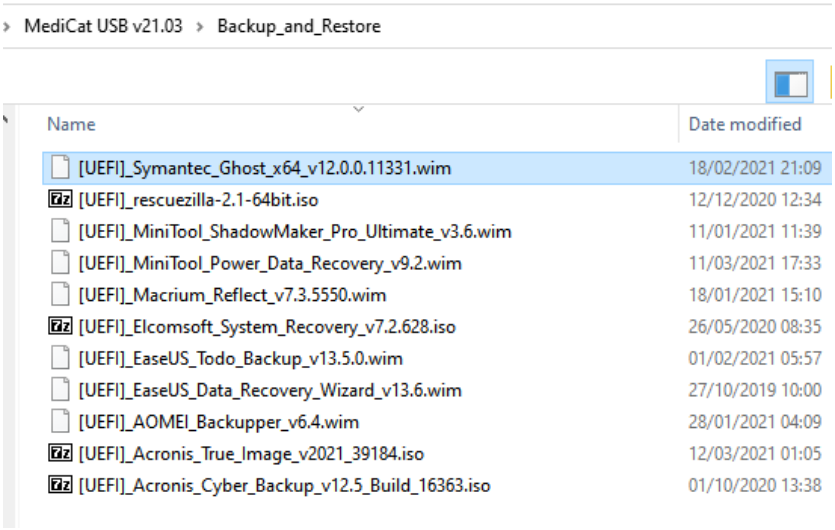
Chapter 15 - Add Windows .wim files (wimboot plugin)

You must add a wimboot plugin file before Ventoy will list and boot from .wim files.

Exercise 17: Add support for booting .wim files

1. Download the **ventoy_wimboot.img** plugin file from https://github.com/ventoy/wimiso/releases and copy the file to the \ventoy folder.
2. Copy your .wim files to the USB drive - e.g. _ISO\WINPE folder.

The Medicat image file download contains many useful .wim files which you can 'borrow'...



You can extract the Symantec_Ghost wim file from the Medicat zip file for instance. Files beginning with [UEFI] should both Legacy and UEFI boot.

Note: Some .wim files may also require other files or folders from Medicat to be added to the USB drive (see here for more details).

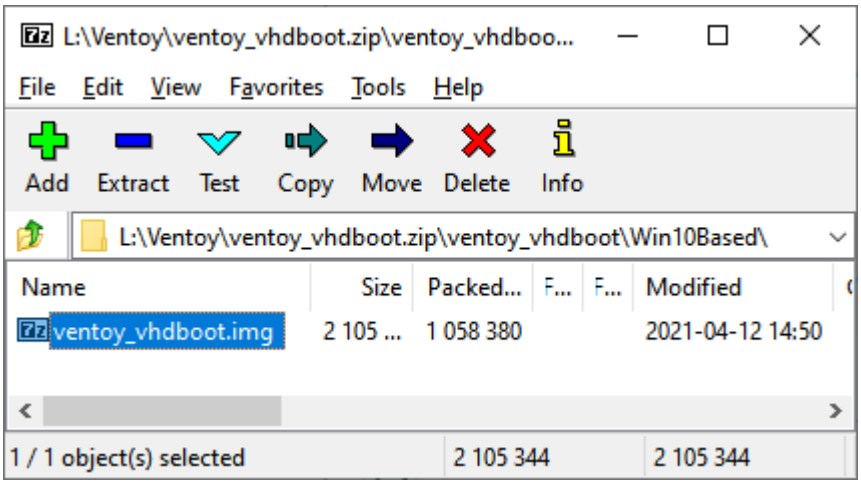
Chapter 16 - Add Windows .VHD files (Win VHD plugin)

Ventoy can boot Windows VHD files, but the process can be a bit temperamental and you may need to experiment with different versions of the plugin. A VHD is a Virtual Hard Disk image. It is basically a copy of the entire contents of a disk but it can be mounted and used as if it was a real disk.

- Ventoy use a plugin file to boot Windows 7+ VHD(x) files.
- Both Legacy BIOS and UEFI are supported.
- Both fixed and dynamic VHD(x) are supported.

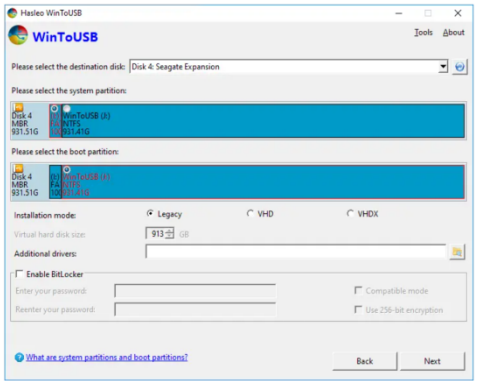
Exercise 18: Add support for .VHD files

1. Download the plugin file '**ventoy_vhdboot.img**' from https://github.com/ventoy/vhdiso/releases and extract the .img file into the \ventoy folder on your USB drive. I used the Win10Based plugin file as shown below...



2. For Windows 10 v1803 and previous versions, Partition 1 must be formatted with an **NTFS** file system, however, for Windows 10 *v1809+* VHD and VHDx files then Partition 1 can be *exFAT or NTFS* (but NTFS is recommended ;-).

Due to Microsoft licensing restrictions, it is difficult to find and download Windows VHD files from the internet (they are quite large anyway!), so you must make your own Windows VHD files. You can use WinToUSB to quickly create a 15-20GB Windows 10 Home VHD file for free (use VHD, and either UEFI or MBR) or install Windows to a VHD manually (see YT video). *Use a separate fast USB drive* with WinToUSB (see here) and make sure you reboot to Windows several times *before* you move the VHD to the Ventoy USB drive, otherwise it will not boot correctly under Ventoy.



< Use the VHD radio button

Chapter 17 - Boot from other drives/partitions/files (grubfm)

The a1ive grub2 file manager (grubfm) is another multiboot USB file manager which is almost as easy to use as Ventoy! An enhanced form of it (called agFM) is used by Easy2Boot, but you can easily use the basic grubfm version just by adding an ISO file to Ventoy.

The advantage is that if Ventoy fails to boot a certain ISO/VHD, you can always try booting to that same payload file using grubfm.

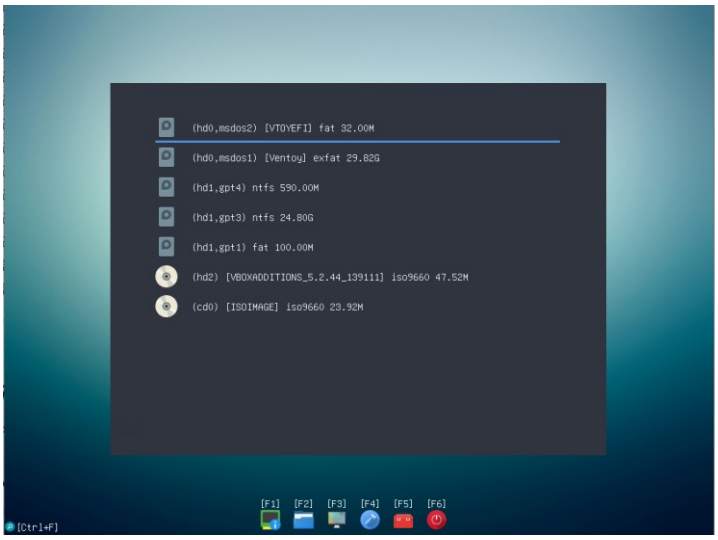
Exercise 19: Add grubfm and boot ISOs from any drive or partition

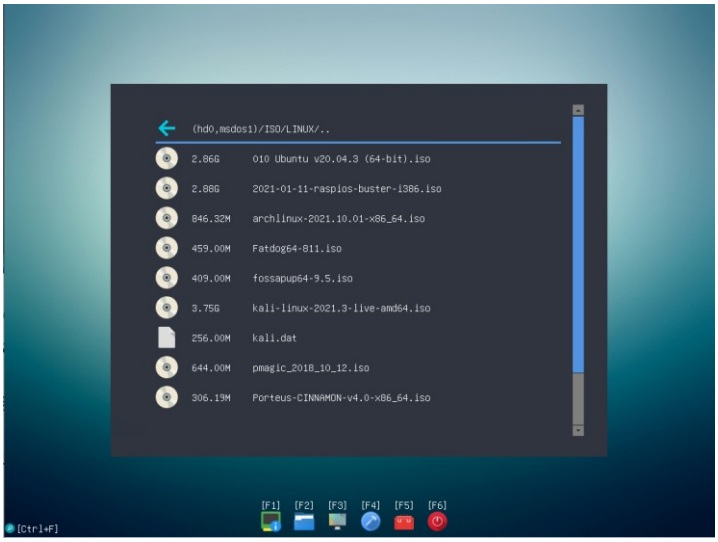
1. Go to the a1ive grubfm GitHub 'Releases' page here and download the latest current release of '**grubfm_multiarch.iso**' (24MB approx.)

2. Copy the ISO file to a suitable folder on your Ventoy USB drive (e.g. \ISO\UTILITIES).

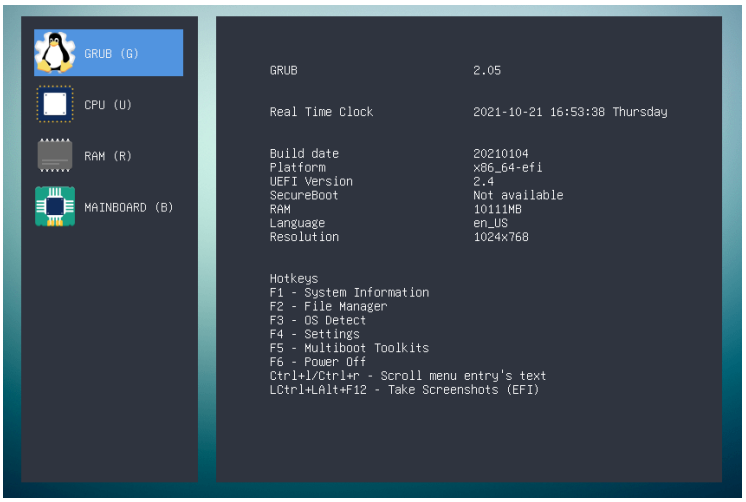
3. Now boot to Ventoy (Legacy\UEFI32\UEFI64) and select the 'grubfm_multiarch.iso' file.

You can now 'explore' to any disk/partition/payload file in the system and boot from it.





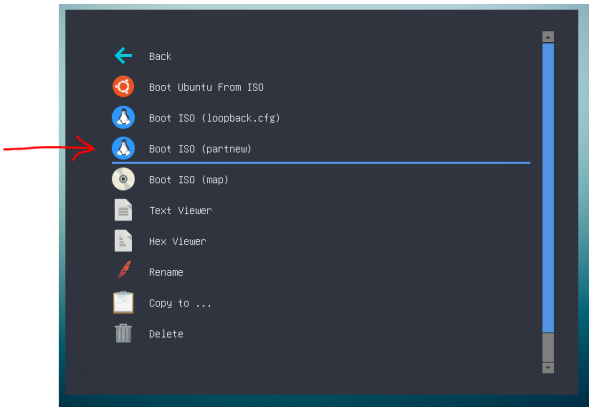
F1 in grubfm shows useful system information:



If you have a large Ventoy USB disk and have created a third partition, you can boot to any payload files that are on Partition 3, as well as all the files on Ventoy's Partition 1.

- F2 File Explorer - press F2 whenever you want to get back to the main explorer menu
- F3 to boot from a different drive/partition.
- F4 to change menu language, keyboard type, resolution, etc.
- F5 will allow you reload Ventoy, load an EFI shell, grub2 console, etc.
- F6 to halt\reboot

Note: grubfm can also uses Easy2Boot's 'partnew' method of booting *Linux ISOs* which is usually successful in 99% of cases.



grubfm offers several boot methods (secondary menu).

However, the ISO file must be made contiguous before grubfm can use the 'partnew' method. You should therefore install 'Defraggler' onto your Windows system and right-click on the ISO file and choose 'Defraggler - Defragment' to ensure it is **contiguous**. The Ventoy USB drive should be **write-enabled** and should contain **Legacy\MBR partitions** with **Primary Partition 4** unused when using the 'partnew' boot menu option.

Tip: WinContig will make all files on a partition contiguous in one go. If you are using a USB HDD (spinning rust drives) then WinContig should also result in speeding up the performance too.

Note that grubfm does not use the \ventoy\ventoy.json file and so persistence and the other Ventoy special functions will not work in grubfm. grubfm does allow you to add .cfg files however and so you can add your

own grub2 menus which can load your .dat persistence file and boot to Linux with persistence, etc. but you will need to know the correct grub2 menu commands for this.

If you are interested in grubfm, you may like to also read eBook #4 from the E2B eBook series:

eBook #4: UEFI-multiboot using the a1ive grub2 File Manager (agFM) How to use the UEFI grub2 based boot manager developed by a1ive to directly Secure UEFI-boot from payloads (e.g. ISO/VHD/EFI/IMG files) without needing to create .imgPTN files in most cases. agFM can also switch-in .imgPTN files from a UEFI-boot or restore the original partitions – no need to boot to WinPE or Windows.

Boot from Partition 3 using Ventoy

The Ventoy menu system only works with payload files on the first partition, but if you have a requirement to place some payload files on Partition 3, then it is possible to use the F6 user grub2 menu extension feature of Ventoy.

Exercise 20: Add StorageCraft Recovery Environment Cross Platform to Partition 3

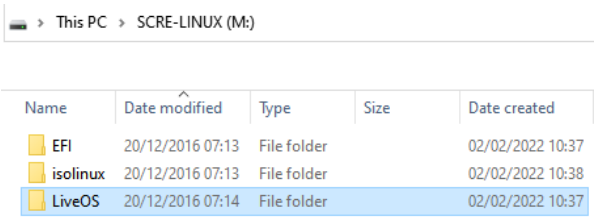
The StorageCraft Recovery ISO is Linux-based and is a commercial product. Although the ISO file boots from Partition 1 using Ventoy, we can extract the files from the ISO file and place them on Partition 3 of the Ventoy USB drive.

Note that if the distro supported an iso-scan/filename type of kernel parameter which allows us to specify the name of the ISO that the kernel should load (like Ubuntu supports), then we would not need to extract the contents of the ISO file!

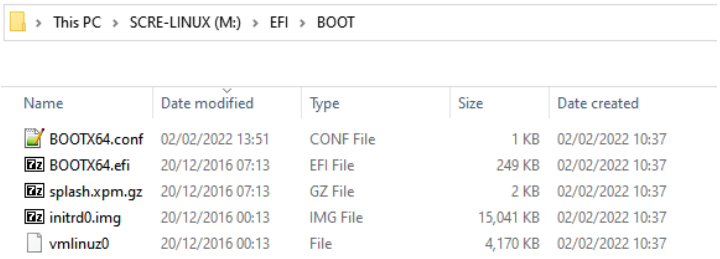
1. First you must resize the partitions on your Ventoy USB drive so that you have a new FAT32 Primary Partition as the third partition. If using 'official Ventoy' you will need to re-make your Ventoy drive and reserve some space at the end of the USB drive so that you can create your third partition after the USB drive has been created. If you are using 'Ventoy for Easy2Boot' you can resize the drive partitions using any utility you like.

2. The Volume Label can be set to **SCRE-linux** or **SCRE-LINUX** (or anything else you like). Note that some utilities will allow you to specify a label in both upper and lower case before the volume is formatted, but Windows will usually force the label to be all upper-case once it has been formatted as FAT32! This means that even though you may type in lower-case for the volume label, it may actually be in all upper-case! Double-check the volume name using Windows Explorer - Properties.

3. Extract the contents of the StorageCraft ISO to the root of partition 3 (the [BOOT] folder is not required if you are using 7zip for the extraction).



4. Edit the **\EFI\BOOT\BOOTX64.conf** text file and change the LABEL= text so to use the same volume label that you have used to Partition 3. Make sure the case is identical!



```
default=0
splashimage=/EFI/BOOT/splash.xpm.gz
timeout 3
hiddenmenu

title SCRE-linux
  kernel /EFI/BOOT/vmlinuz0 root=live:LABEL=SCRE-LINUX rootfstype=auto ro liveimg quiet selinux=0 rhgb
  initrd /EFI/BOOT/initrd0.img
```

Note: It is very unusual to see this type of syntax used as it is not the usual grub2 syntax!

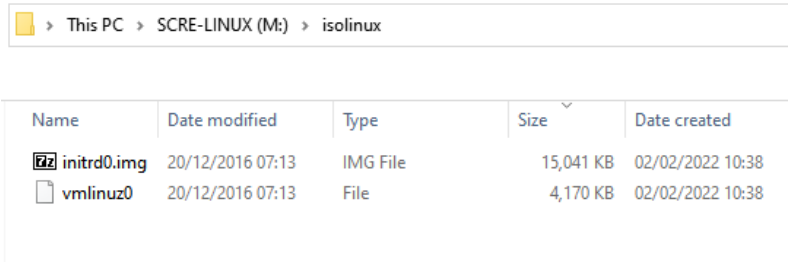
5. Now make (or add) some menu entries to your \ventoy\ventoy_grub.cfg text file on Partition 1.

Download ventoy_grub.cfg here.

The contents are shown below (note that some lines may wrap over to the next line):

```
#/ventoy/ventoy_grub.cfg file for booting StorageCraft Recovery Environment Cross Platform from Ventoy Partition 3
#Make FAT32 Partition 3 with volume label of "SCRE-linux"    (may be case sensitive - note that Windows may force the label to
all upper-case!)
#Extract contents of the ISO to root of Partition 3
#If you use a different label for Ptn3 - e.g. SCRE-LINUX, then change the LABEL= text in this file and in the
\EFI\BOOT\BOOTX64.conf file (case-sensitive)

if [ "${grub_platform}_${grub_cpu}" = "pc_i386" ]; then
          menuentry "StorageCraft Recovery Environment Cross Platform" {
            set root=(${vtoydev},3)
            linux16 /isolinux/vmlinuz0 root=live:LABEL=SCRE-LINUX rootfstype=auto ro liveimg quiet selinux=0 rhgb rd_NO_LUKS
rd_NO_MD rd_NO_DM
            initrd16 /isolinux/initrd0.img
          }
          menuentry "StorageCraft Recovery Environment Cross Platform (Basic video)" {
            set root=(${vtoydev},3)
            linux16 /isolinux/vmlinuz0 root=live:LABEL=SCRE-LINUX rootfstype=auto ro liveimg quiet selinux=0 rhgb rd_NO_LUKS
rd_NO_MD rd_NO_DM xdriver=vesa nomodeset
            initrd16 /isolinux/initrd0.img
          }

else
          if [ "$grub_cpu" = "x86_64" ]; then
                    menuentry "StorageCraft Recovery Environment Cross Platform (UEFI64)" {
                      set root=(${vtoydev},3)
                      chainloader /EFI/BOOT/BOOTX64.efi
                    }
          fi
fi

menuentry '<-- Return to previous menu [Esc]' --class=vtoyret VTOY_RET {
    echo 'Return ...'
}
```

The commands I used for Legacy booting were converted from the menus in the \isolinux\isolinux.cfg file. Only the two files below are required for legacy-booting.



Note that for UEFI, I have chainloaded the .EFI boot file instead of using linux and initrd commands (or linuxefi and initrdefi commands). This was because, in this case, even though the grub2 Linux and initrd commands worked under grubfm/agFM, then could not be made to work under Ventoy!

6. You can now boot to Ventoy and press F6 for the external menu and boot to the StorageCraft Linux OS.

Note: If using an Easy2Boot USB drive and 'Ventoy for Easy2Boot', after resizing the partitions you may find that .imgptn files will no longer work because the backup partition sector at LBA30 will no

longer match the MBR. In this case just Legacy-boot to the E2B menu, press shift+c to get to the grub4dos command console and type `fix30` *to reset the backup sector at LBA30.*

grub2 menu entries for ISO file booting

There are many examples of how to boot directly from an ISO file using grub2 on the internet. They nearly always involve specifying the path and name of the ISO file in the kernel parameters.

```
menuentry "Ubuntu Live CD" {
    set isofile="/ubuntu.iso"
    loopback loop (hd0,msdos1)$isofile
    linux (loop)/casper/vmlinuz boot=casper iso-scan/filename=$isofile noprompt noeject debug
    initrd (loop)/casper/initrd
}
```

Note: Avoid using spaces in filenames because most kernel parameter values are not able to cope with spaces and they cannot be quoted or 'escaped' - this is why 99% of all Linux ISOs do not have spaces in their filenames!

We would convert this for Ventoy and Partition 3 like this:

```
set BIT64=0
if cpuid -l; then set BIT64=1;fi

if [ "${grub_platform}_${grub_cpu}" = "pc_i386" ]; then
# legacy menu here for 64-bit kernel
          if [ "$BIT64" = "1" ]; then
                    menuentry "Ubuntu Live CD ISO on Partition 3 (Legacy 64-bit)" {
                              set isofile="/LINUX/ubuntu.iso"
                              loopback loop (${vtoydev},3)$isofile
                              linux (loop)/casper/vmlinuz boot=casper iso-scan/filename=$isofile noprompt noeject debug
                              initrd (loop)/casper/initrd
                    }
          fi
else
          if [ "$grub_cpu" = "x86_64" ]; then
                    menuentry "Ubuntu Live CD ISO on Partition 3 (UEFI64)" {
                              set isofile="/LINUX/ubuntu.iso"
                              loopback loop (${vtoydev},3)$isofile
                              linux (loop)/casper/vmlinuz boot=casper iso-scan/filename=$isofile noprompt noeject debug
                              initrd (loop)/casper/initrd
                    }
          fi
fi
```

The iso file should be in the LINUX folder on partition 3.

Note that the kernel parameters vary from version to version and from distro to distro! You will need to check the .cfg files that can be found within each ISO. Tip: Look for a /boot/grub/loopback.cfg file as these usually contain the keywords used to specify the ISO file.

Many Linux distros do not allow you to specify the ISO file path and name - you must check the documentation!

Note also that some kernels do not support certain file systems and so they may not be able to find your ISO. For instance, if the ISO is on an NTFS partition then the Linux boot kernel must have been compiled with integrated NTFS file system drivers or else it will not be able to access the volume that contains your ISO even though Ventoy could access it. Some kernels may only contain ISO9660 CD filesystem drivers and will not be able to access your disk volume at all!

Kasperksy Rescue Disk ISO on Partition 3

The ISO MUST be in the root folder **\data** (e.g. \data\krd.iso).

Both FAT32 and NTFS volumes work. Legacy and UEFI64 work. Only tested on MBR-partitioned USB drive (not GPT).

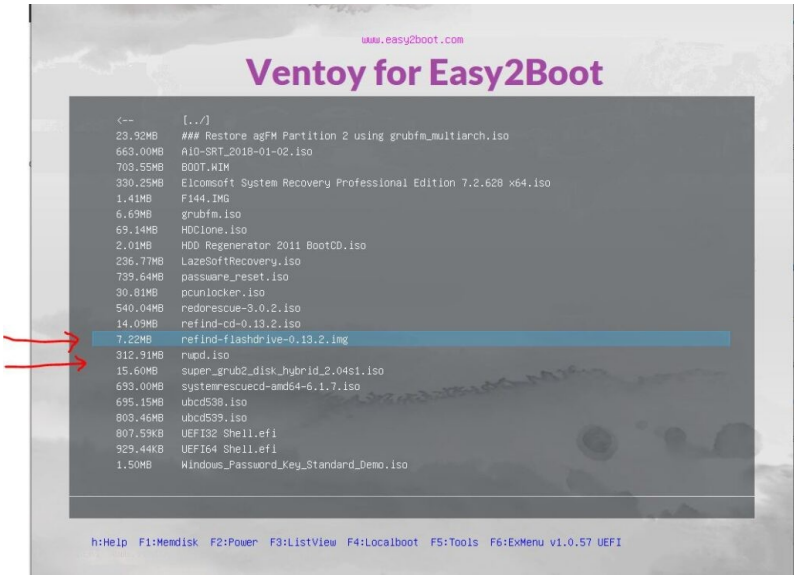
If you change the name of the ISO you must also change the menu references in red.

```
# ISO file must be located at /data/krd.iso

menuentry "Kaspersky Rescue Disk 2018 (PTN 3)" {
          set iso_path="/data/krd.iso"
          set lang="en"
          set root=(${vtoydev},3)
          # en=English; ru=Russian
          if ! [ "x${lang}" = "xru" ]; then set lang=en; fi
          echo Booting (${root})${iso_path}  $lang
          #search -s root -f ${iso_path}
          loopback loop ${iso_path}
          if cpuid -l; then set _kernel="k-x86_64"; else set _kernel_="k-x86"; fi
          linux     (loop)/boot/grub/${_kernel} net.ifnames=0 lang=${lang} dostartx isoloop=krd.iso
          initrd    (loop)/boot/grub/initrd.xz
}
```

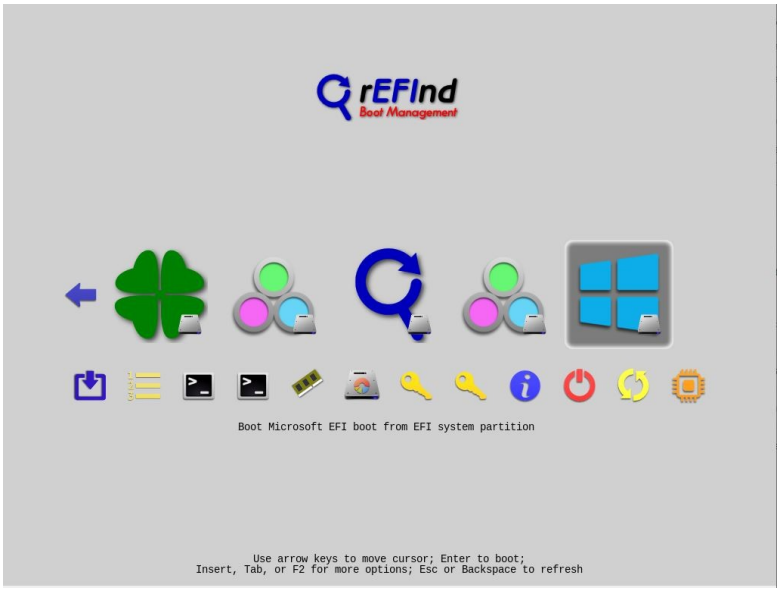
If you uncomment the 'search' line then it should find the iso on any valid volume in the system (in /data folder), so if you have more than one krd.iso then it may not load the one you wanted!

Chapter 18 - Add rEFInd to UEFI-boot from any disk/partition

rEFInd is a UEFI boot manager. First you must UEFI64-boot to agFM or Ventoy and then you can run rEFInd which will allow you to load .EFI files from other disks and partitions in the system. This is especially useful for Ventoy USB drives because Ventoy only enumerates the .EFI boot files which are on the Ventoy USB drive's first partition.



You can add both the rEFInd .iso file and the .img file, but the .img fle seems to work best.

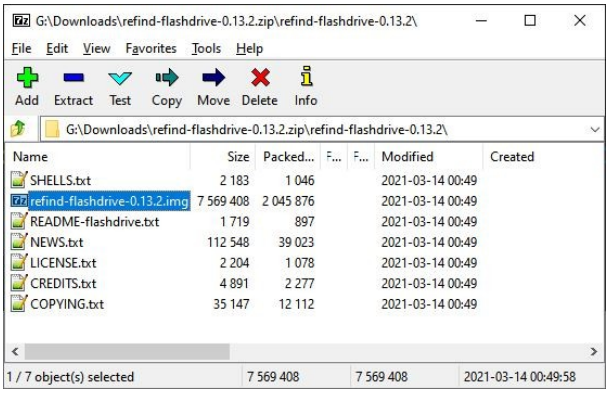


The rEFInd USB .img file displays many UEFI boot options.

Exercise 21: Add rEFInd.img

Download the 'A USB flash drive image file' .zip file from here.
(The .iso CD-R image file but it does not seem to work as well so don't use the .iso file).

Use 7Zip to extract the .img file.



Extract the .img file.

3. Copy the .img file to the first partition of your E2B\agFM\Ventoy USB drive (place in any suitable folder).

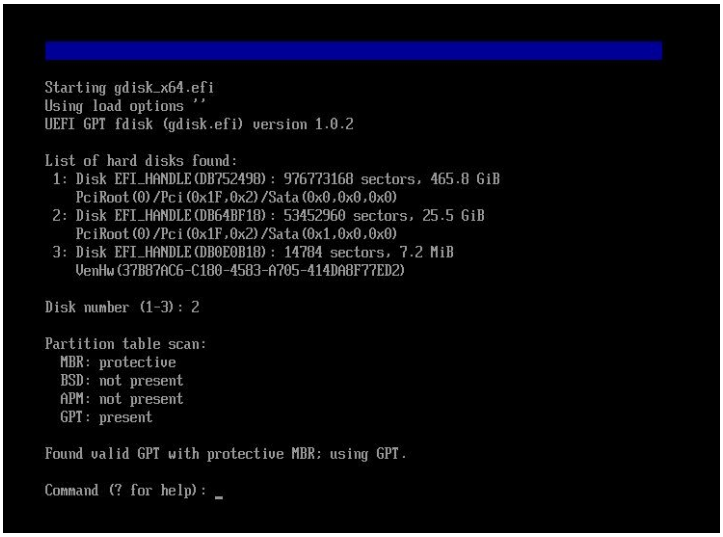
Usage

You must UEFI-boot to Ventoy because rEFInd is a UEFI64 utility and must run from a UEFI64 environment.

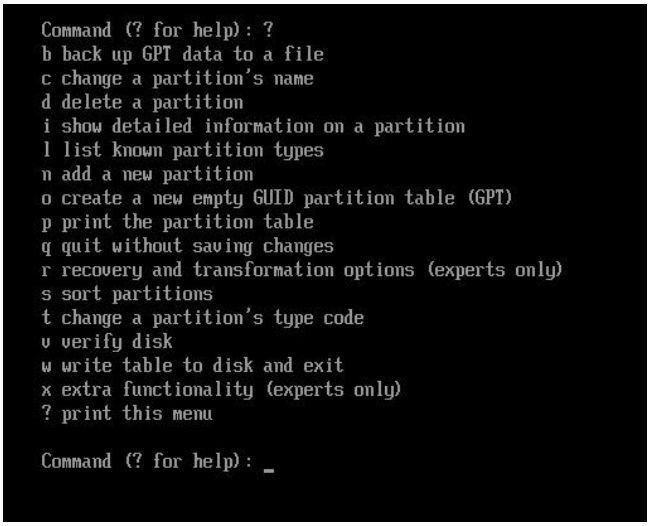
Once rEFInd is loaded, it should search all drives on the system and list all UEFI64-bootable files. This allows you to boot from .EFI files and from UEFI64-bootable partitions.

Recent versions can also identify tools such a MokManager and MemTest86. You can also load an EFI shell, reboot to the internal EFI shell, install rEFInd to a disk, manage the BIOS, EFI boot order of the system, run a disk partitioning tool (gdisk_x64.efi – see below), reboot to BIOS setup, etc.

The image also contains gdisk which can be used for partitioning a drive under UEFI:



Disk partitioning tool (gdisk).

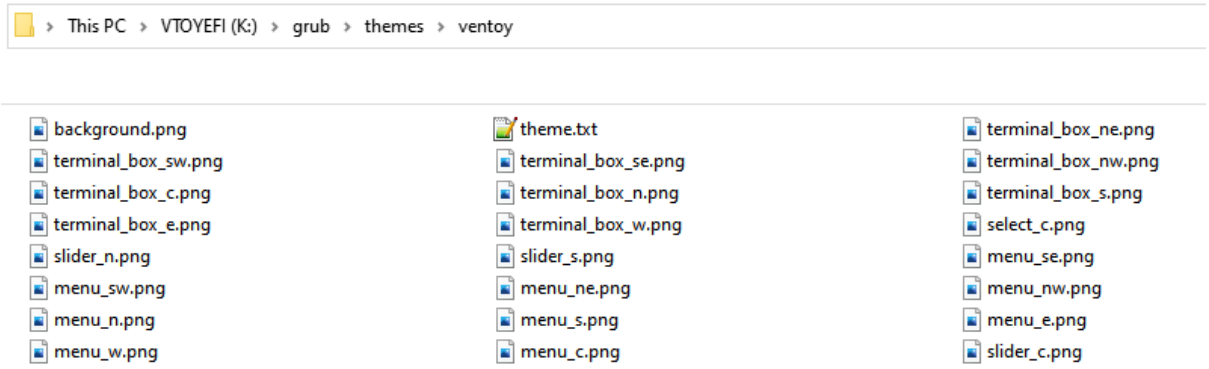


The commands in gdisk.

Chapter 19 - Change the default Ventoy menu theme

Ventoy is based on grub2 which can operate in text mode (press F7 in the Ventoy menu to see what that looks like) or in Graphical mode (Ventoys default mode).

When in graphical mode, grub2 uses a 'Theme'. The default theme used by Ventoy is 'built-in', but you can see the files used by looking in the \grub\themes\ventoy folder on Partition 2:



The current source files can also be viewed here (I have also pasted the theme.txt file below).

You must not change any files on Partition 2 however, because when you update Ventoy with a newer version, all your changes will be lost.

The correct way to modify or change the theme is to place your own theme files under a new folder in the \ventoy folder of Partition 1 and then add a theme entry into your \ventoy\ventoy.json file.

Most themes contain a file called **background.png** which is the background wallpaper used by that theme and the **theme.txt** file is the theme configuration file. Other .png files may be used for borders, boxes, sliders, etc.

Some themes also include font files so that text on the menu can be displayed in different fonts, and a theme may also contain an *icons* folder which contains small .png files which are displayed next to the menu entries.

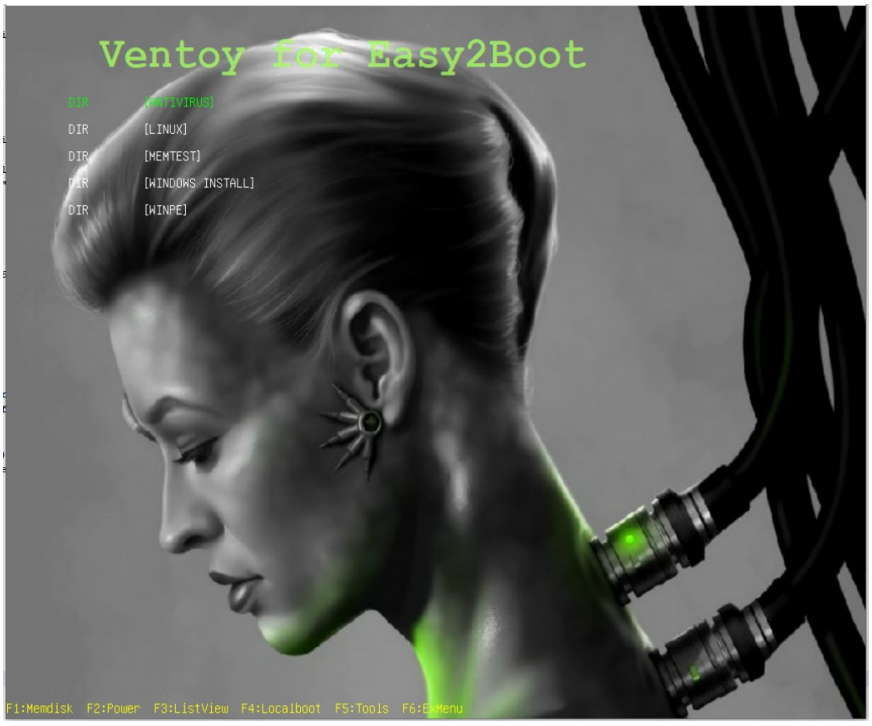
E2B eBook #4 contains more details on how grub2 themes work.

Ventoy v1.0.57 'theme.txt'

```
desktop-image: "background.png"
title-text: " "
title-color: "#ffffff"
message-color: "#f2f2f2"

terminal-box: "terminal_box_*.png"

+ boot_menu {
  left = 10%
  width = 80%
  top = 30%
  height = 50%

  menu_pixmap_style = "menu_*.png"

  item_color = "#ffffff"
  item_height = 30

  item_spacing = 1
  item_padding = 1

  selected_item_color= "#f2f2f2"
  selected_item_pixmap_style = "select_*.png"

  item_icon_space = 0

  scrollbar = true
  scrollbar_width = 10
  scrollbar_thumb = "slider_*.png"
}



+ progress_bar {
  id = "__timeout__"
  text = "@TIMEOUT_NOTIFICATION_SHORT@"

  left = 90%
  width = 10%
  top = 90%

  text_color = "red"
  bar_style = "*"
  highlight_style = "*"
}

+ hbox{
    left = 28%
    top = 95%
    width = 10%
    height = 25
    + label {text = "@VTOY_HOTKEY_TIP@" color = "blue" align = "left"}
}


+ hbox{
    left = 30%
    top = 95%-25
    width = 10%
```

```
    height = 25
    + label {text = "@VTOY_MEM_DISK@" color = "red" align = "left"}
}


+ hbox{
    left = 30%
    top = 95%-50
    width = 10%
    height = 25
    + label {text = "@VTOY_ISO_RAW@" color = "red" align = "left"}
}


+ hbox{
    left = 30%+200
    top = 95%-50
    width = 10%
    height = 25
    + label {text = "@VTOY_GRUB2_MODE@" color = "red" align = "left"}
}

+ hbox{
    left = 30%+200
    top = 95%-25
    width = 10%
    height = 25
    + label {text = "@VTOY_WIMBOOT_MODE@" color = "red" align = "left"}
}

+ hbox{
    left = 90%
    top = 55
    width = 10%
    height = 25
    + label {text = "@VTOY_ISO_UEFI_DRV@" color = "red" align = "left"}
}
```

Exercise 22: Add a 'Seven-of-nine' theme



1. Download the Ventoy_Seven_20_1.55_theme_eBook.zip file from here.

This theme allows us to show twenty entries per page on the screen (useful if you have lots of ISO files and a large USB drive).

2. Extract the contents of the .zip file to the \ventoy folder on Partition 1.

You should now have these new folders:

- \ventoy\themes\
- \ventoy\themes\fonts\
- \ventoy\themes\e2b_seven_20\

3. Edit your \ventoy\ventoy.json file to add in the following theme section:

```
"theme": {
    "file": [
      "/ventoy/themes/e2b_seven_20/theme.txt"
    ],
      "gfxmode": "1024x768",
      "display_mode": "GUI",
      "serial_param": "--unit=0 --speed=9600",
      "ventoy_left": "5%",
      "ventoy_top": "95%",
      "ventoy_color": "#7d7d7d",
      "fonts": [
                "/ventoy/fonts/terminus-b18.pf2"
               ]
}
```

There is a sample ventoy.json text file included in the zip file (Seven_20_sample_ventoy.json.txt) which you can copy and paste into your ventoy.json file if you have problems editing your ventoy.json file.

Note that this theme uses a .pf2 font file, so we must load the correct font file using a "fonts" key in the Ventoy.json file. Some themes may use more than one font file and so you must specify them all in the JSON file.

The graphics resolution is set at 1024x768. This is the standard default resolution for the UEFI BIOS shell and is therefore the most compatible resolution to use because it should be supported by all UEFI systems and monitors.

The 'display_mode' is set to GUI mode because grub2 themes only work in this mode.

The 'serial_param' key is only valid if you are using a serial console in grub2 so for most people that line can be ignored or deleted.

The 'ventoy_left', 'ventoy_top' and 'ventoy_color' keys in ventoy.json define the location and colour of the **Ventoy version text**. If you do not want it to be displayed, then you can position the text to somewhere on the screen which has a uniform colour and then set the ventoy_color to be the same colour as the background (for instance, if there is a black background then set the ventoy_color to #000000 which is black).

Transparent colour: In grub2 (and any theme.txt file used by Ventoy) the hexadecimal colour value can consist of three *or four* hexadecimal bytes (i.e. either #RRGGBB or #RRGGBB**AA** where AA is the Alpha transparency value where 00 = completely transparent and FF = opaque/solid). So a colour value of #112233**00** will make the colour completely transparent and thus invisible because it ends on 00 (any value can be used instead of 112233). This means you can set the ventoy_color value to transparent in the ventoy.json file by adding 00 to any colour value:

```
"theme":{
    "file":[
        "/ventoy/themes/default/theme.txt",
    ],
    "ventoy_color": "#ff000000"
},
```

Tip: If you set VTLE_CLR to any colour value by editing the Ventoy /grub/grub.cfg file then it will permanently set the Ventoy version string to that colour (the ventoy_color setting in ventoy.json will have no effect). e.g. for no version string, use:

```
set VTLE_CLR=#ffff0000
```

If you want to change the background picture, simply change the \ventoy\themes\e2b_seven_20\seven.jpg file. Note that you can have a picture with a transparent background but it must be a .png file. For more details about transparency and .png files, see this article.

The text **'Ventoy for Easy2Boot'** comes from the file **logoVforE2B.png** and it is defined in the **theme.txt** file. It can have a transparent background so that it can be overlaid onto the background wallpaper. You can delete that section from the theme.txt file if you do not want the logo:

```
# Show logo
```

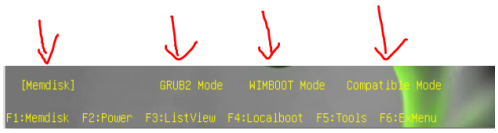
```
+ image {
 id = "__my_logo__"
 left = 10%
 top = 1%
 width = 4%
 height = 6%
 file = "logoVforE2B.png"
}
```

or you can change it to use your own logo file.

4. In the theme.txt file you will see various 'hbox' sections. These are responsible for displaying the following text strings:

- **@VTOY_HOTKEY_TIP@** - " h:Help F1:Memdisk F2:Power F3:TreeView F4:Localboot F5:Tools F6:ExMenu "
- **@VTOY_MEM_DISK@** - "[Memdisk]"
- **@VTOY_GRUB2_MODE@** - "GRUB2 Mode"
- **@VTOY_WIMBOOT_MODE@** - "WIMBOOT Mode"
- **@VTOY_ISO_RAW@** - "Compatible Mode"
- **@VTOY_ISO_UEFI_DRV@** - "UEFI FS" - *Ctrl+u (debug only)*

Try pressing Ctrl+l, Ctrl+r, Ctrl+w and F1 (and Ctrl+u) to see the text displayed on the screen:



Exercise 23: Random themes

Ventoy allows you to specify multiple themes and it will randomly choose a different theme each time you boot.

1. Download the **ventoy_json_example_with_6_different_themes_ebook.zip** file from here

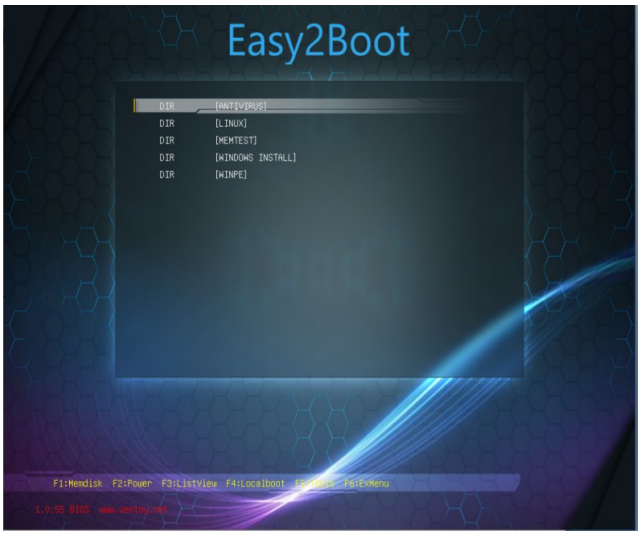
2. Extract the files inside to your Ventoy drive Partition 1. Note that **this download contains a \ventoy\ventoy.json file,** so make sure you back up your own Ventoy.json file first before you overwrite it!

Now each time you reboot, you should see one of the six different themes.

You can add the same theme lines into your own Ventoy.json file:

```
"theme": {
    "file": [
      "/ventoy/themes/ventoy/theme.txt",
      "/ventoy/themes/default/theme.txt",
      "/ventoy/themes/desktop/theme.txt",
      "/ventoy/themes/honeycomb/theme.txt",
      "/ventoy/themes/litemint/theme.txt",
      "/ventoy/themes/Ventoy-Dark/theme.txt"
    ],
      "gfxmode": "1024x768",
      "display_mode": "GUI",
      "serial_param": "--unit=0 --speed=9600",
      "ventoy_left": "5%",
      "ventoy_top": "95%",
      "ventoy_color": "red",

    "fonts": [
    "/ventoy/fonts/terminus-b18.pf2"
    ]
}
```

If you just want to try *one* of the themes, simply change the filenames of the other five theme.txt files to a name which does not exist (e.g. themeX.txt). Here is the honeycomb theme...



The 'Easy2Boot' logo comes from a logo.png file in each theme folder - change it or delete the file if you don't want any logo.

Tip: Ventoy 1.0.62 and later allows you to choose any of the themes using F5 - Theme Select. You can also use 'default_file' in the ventoy.json file to select one of the themes on start-up:

default_file	INTEGER	This option only take effect when 'file' key contains more than one them. 0: randomly select a theme on boot (this is the default value). 1: Select the 1st theme on boot. 2: Select the 2nd theme on boot. 3: Select the 3rd theme on boot. etc.

Exercise 24: How to make your own Ventoy theme

If you find a grub2 theme that you like, you can use it with Ventoy but you will need to change the theme.txt file.

You can look here for some ready-made themes. Some even have a Ventoy variant (although they are often missing the extra 'hbox' elements and so the 'secret' hotkey modes or the bottom Ventoy F-key help text is not displayed).

Here is the procedure to follow to convert a grub2 theme:

1. Download a grub2 theme that you like and copy the folder to your \ventoy\themes folder.

2. Edit the themes.txt file to add the hbox elements used by Ventoy (see the default theme.txt above).

You may need to adjust the position and colour of the various elements by editing the settings in the theme file.

3. If any fonts are specified in the theme.txt file, make sure they are present on your Ventoy USB drive. Normally the .pf2 font files are in the same folder as the theme files.

4. Now edit your \ventoy\ventoy.json file to add in your theme. Don't forget to specify all font files too.

Now test the theme by booting (either using a Virtual Machine, QEMU or a real system).

Don't forget to press F1, Ctrl+i, Ctrl+r, Ctrl+u and Ctrl+w (and h) to test the *hbox* elements too.

You can find the current Ventoy default theme.txt file here.

The grub2 theme.txt format is detailed in the grub2 manual here if you want to go down that rabbit hole!

The theme.txt menu can position elements using pixels (plain numbers) and\or as a percentage of the screen width. If you want to use more than one resolution mode (or change resolution within the Ventoy menu system using the F keys) then it is best to use percentages.

Chapter 20 - Specify the exact menu entries (Image List plugin)

This feature allows you to define exactly what will be in the Ventoy menu and in what order, or you can just hide specific payload files but show all the others.

In the \ventoy\ventoy.json file you can define ONE of the following two keys:

image_list - a list of which payloads to show.
image_blacklist - a list of which payloads to NOT show.

We can also add '_legacy' and '_uefi' to the key name (see multimode page for more details).

Exercise 25: Exclude legacy-only ISO files in the UEFI menu

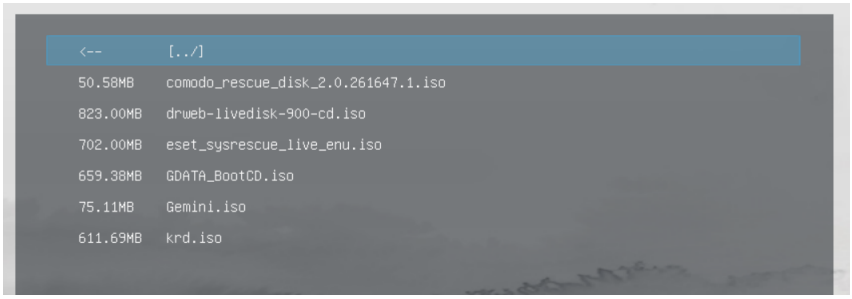
In the AntiVirus Chapter, the Comodo ISO only worked if we Legacy-boot because it does not support UEFI. We can exclude that ISO from being listed by Ventoy when we UEFI-boot.

The screenshots below show the ANTIVIRUS menu in Legacy mode and in UEFI mode after the Ventoy.json file had been modified to contain an **image_blacklist** entry.

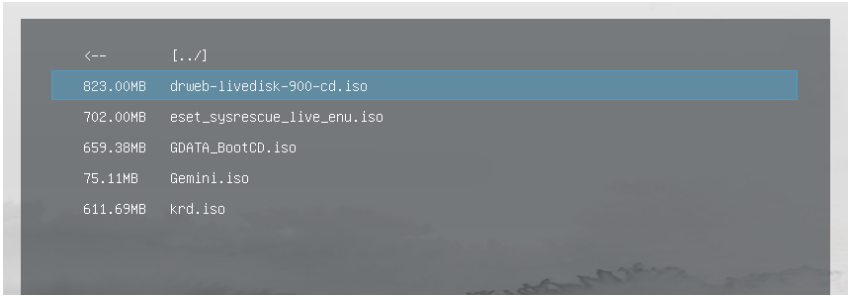
I simply added a section to the bottom of the Ventoy.json file so that the Comodo ISO file is not listed whenever I UEFI-boot from the Ventoy USB drive:

```
"image_blacklist_uefi": [
      "/ISO/ANTIVIRUS/comodo_rescue_disk_2.0.261647.1.iso"
]
```

Legacy Menu:



UEFI Menu (notice that the comodo ISO file is not listed):



The full JSON file is shown below (with three blacklisted files):

```
{
      "control": [{
                  "VTOY_DEFAULT_MENU_MODE": "1"
            },
            {
                  "VTOY_FILT_DOT_UNDERSCORE_FILE": "1"
            },
            {
                  "VTOY_TREE_VIEW_MENU_STYLE": "0"
            },
            {
                  "VTOY_DEFAULT_SEARCH_ROOT": "/ISO"
            },
            {
                  "VTOY_DEFAULT_KBD_LAYOUT": "QWERTY_UK"
            }
      ],
      "persistence": [{
            "image": "/ISO/LINUX/010 Ubuntu v20.04.3 (64-bit).iso",
            "backend": [
                  "/ISO/LINUX/ubuntu512ext4.dat",
                  "/ISO/LINUX/ubuntu512xfs.dat"
            ]
      }, {
            "image": "/ISO/LINUX/Ubuntu-v20.04.3.iso",
            "backend": "/ISO/LINUX/ubuntu512xfs.dat"
      }],

      "auto_install_legacy": [{
                  "image": "/ISO/WINDOWS
INSTALL/Win10_EnglishInternational_x64.iso",
                  "template": [
                        "/ISO/WINDOWS INSTALL/Win10ProLegacy.xml",
                        "/ISO/WINDOWS INSTALL/Win10ProLegacy2.xml"
                  ]
            },
            {
                  "image": "/ISO/WINDOWS
INSTALL/Win11_EnglishInternational_x64.iso",
                  "template": [
                        "/ISO/WINDOWS INSTALL/Windows 11 Home fix tpm.xml",
                        "/ISO/WINDOWS INSTALL/Windows 11 Pro fix tpm.xml"
                  ]
            }
      ],

      "auto_install_uefi": [{
                  "image": "/ISO/WINDOWS
INSTALL/Win10_EnglishInternational_x64.iso",
                  "template": [
                        "/ISO/WINDOWS INSTALL/Win10ProUEFI.xml",
                        "/ISO/WINDOWS INSTALL/Win10ProUEFI2.xml"
                  ]
            },
            {
                  "image": "/ISO/WINDOWS
INSTALL/Win11_EnglishInternational_x64.iso",
                  "template": [
                        "/ISO/WINDOWS INSTALL/Windows 11 Home fix tpm.xml",
                        "/ISO/WINDOWS INSTALL/Windows 11 Pro fix tpm.xml"
                  ]
```

```
            }
      ],
      "image_blacklist_uefi": [
            "/ISO/ANTIVIRUS/comodo_rescue_disk_2.0.261647.1.iso",
            "/ISO/WINDOWS
INSTALL/windows_10_enterprise_ltsc_2019_x86_dvd.iso",
            "/ISO/WINPE/Win10PE_x86.iso"
      ]
}
```

Whitelist menu

Instead of a blacklist, You can use "image_list" to define exactly what order the files will be listed in and which files you want to be listed. This would perhaps be most appropriate if your menu is in List View mode and you only want a single screen full of payloads to be displayed in a specific order.

Here is a small Ventoy.json file for displaying a very small menu in Ventoy. The UEFI menu will have only two menu entries and the Legacy menu will have two menu entries.

```
{
	"image_list_uefi": [
		"/ISO/WINDOWS INSTALL/Win11_EnglishInternational_x64.iso",
		"/ISO/WINPE/HBCD_PE_x64_102.iso"
	],
	"image_list_legacy": [
		"/ISO/ANTIVIRUS/comodo_rescue_disk_2.0.261647.1.iso",
		"/ISO/WINPE/HBCD_PE_x64_102.iso"
	]
}
```

Chapter 21 - Replace the menu filenames with text (Alias plugin)

You are, of course, able to change the file name of your ISO just be renaming the file. However, the file names are sorted alphabetically in the Ventoy menu, so if you wanted to preset their order in the menu, you could use a numbering system to make the Ubuntu ISO file which begins with 'U' appear first in the menu, but then the menu would look like this after renaming the files:

010 Ubuntu_v2.0.3_x64.iso
020 Arch_Linux.iso
030 Linux Mint x64.iso

If you do not want the numbers to be listed in the menu (but you want to control the item order), you can set an 'Alias' name for each file in the Ventoy.json file.

Using an 'Alias' also allows you to use characters which may not be allowed in Windows or Linux filenames such as `: & $ ? * / < > |` .

So by using Aliases, the Ventoy menu could now display:

Ubuntu 64-bit (v2.0.3) <UEFI>
Arch Linux 64-bit (may not work on PC3?)
Linux Mint 64-bit (50:50)

You can use the 'image' and 'dir' keys to set the menu entry strings for each one.

Note that for non-ANSII characters (e.g. with letters with accents, cedillas, umlauts, Chinese characters, etc.), you must save the Ventoy.json file in UTF-8 format. In Windows Notepad this is easily done by using 'File - Save as' and setting the Encoding to UTF-8:

Here is an example menu_alias section in a Ventoy.json file:

```
{
    "menu_alias": [
        {
            "image": "/ISO/LINUX/MX-19.1_x64.iso",
            "alias": "MX 19.1 ISO file For me"
        },
        {
            "image": "/ISO/LINUX/ubuntu-20.04-desktop-amd64.iso",
            "alias": "Ubuntu +++ <2004> +++"
        },
        {
            "image": "/ISO/WINDOWS
INSTALL/_windows_10_enterprise_ltsc_2019_x64_dvd_9c09ff24.iso",
            "alias": "我的 Windows 10 系统"
        },
        {
            "dir": "/ISO/LINUX",
            "alias": "[My Linux Directory]"
        }
    ]
}
```

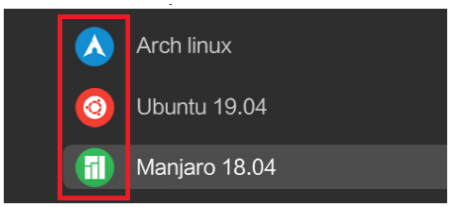
For more information, refer to the Ventoy Alias plugin page here.

Chapter 22 - How to display grub2 menu icons (Class plugin)

If you have used an "image_list" (see previous Chapter) you may also wish to display a nice icon next to each menu entry.

To do this you will need to specify your own theme in the Ventoy.json file.

The icons folder in your theme folder must contain all the .png icon files that you want to use (you can also use any of the built-in Ventoy icon files). More details here.



In the grub2 menu system, the **--class** parameter specifies the name if the icon file to be displayed, so in Ventoy these entries are named "class".

Key	Type	Description
key	STRING	The key string. Ventoy will do a substring match (case sensitive) with the ISO name and this key. If this key is a substring of the ISO name, then the ISO menu will use the class. Only the name of the ISO file will be used for match (not the full path)
parent	STRING	The full path of a directory. Ventoy will use the class for all the image files under this directory and its subdirectories. Must NOT end in /
dir	STRING	The full path of a directory. Ventoy will do a full string match (case sensitive) and use the class if matched. Must not end in /
class	STRING	menu class string (max length: 128) - filename of icon (no file extension)

For instance, if you keep all your Ubuntu ISOs in one folder, you could use the dir key as follows:

```
{
	"menu_class": [{
		"dir": "/ISO/LINUX/UBUNTU",
		"class": "linux"
	}]
}
```

The .\icons\linux.png file in your theme folder will be displayed next to any file in the UBUNTU folder.

Note that the entries are case-sensitive.

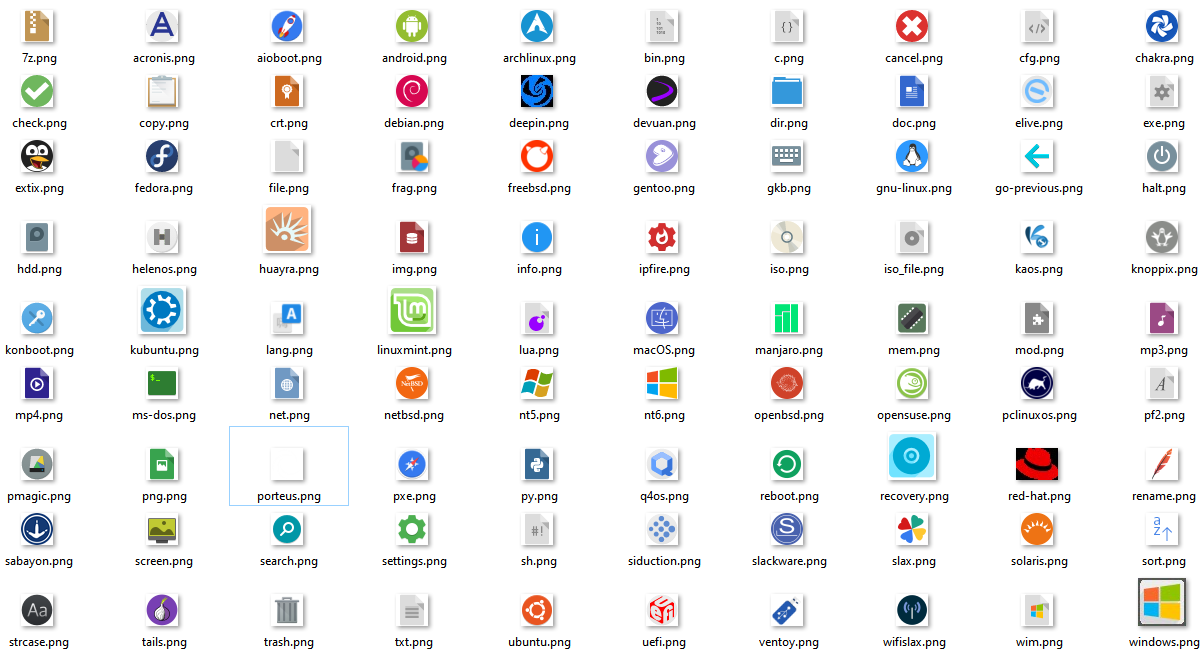
If you wanted all Ubuntu ISOs to be displayed with an ubuntu icon no matter what folder they were in, you could use:

```
{
	"menu_class": [{
```

```
            "key": "buntu",
            "class": "ubuntu"
        }]
}
```

The ubuntu.png file must be present in your theme folder inside the icons folder. Any file with the lowercase letters 'buntu' will be displayed with the ubuntu.png icon on the l-hand side of the menu entry.

Many grub2 themes also contain .png icon files, you can **download** a grub2 zip file containing these 32x32 icons here.

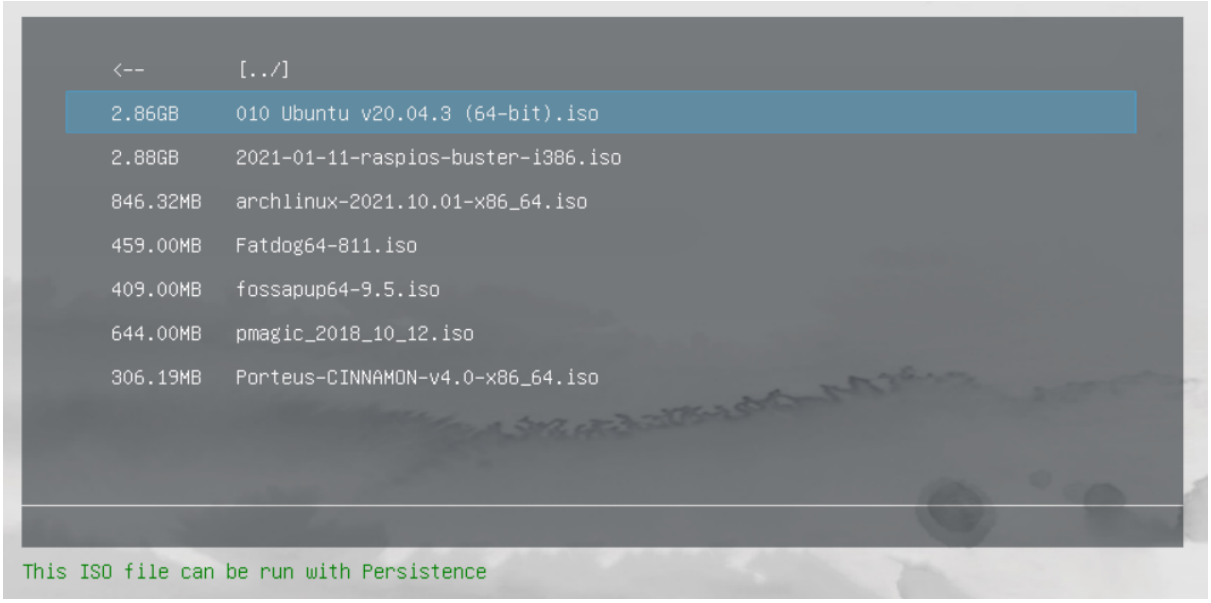


Chapter 23 - Display payload tips to the user (MenuTip plugin)

Recent versions of Ventoy allow you to display a tip for each menu item.

For instance, you may have to press Ctrl+i before booting to a certain ISO, otherwise it will fail to boot or you may need to press F7 for text mode before you boot to a particular WinPE ISO. Perhaps you just want to give some information to the user about what that menu item will do, etc.

Here is an example of a tip in green text for the Ubuntu ISO file which is displayed just below the standard Ventoy menu area:



More details can be found on the Ventoy MenuTip plugin page here.

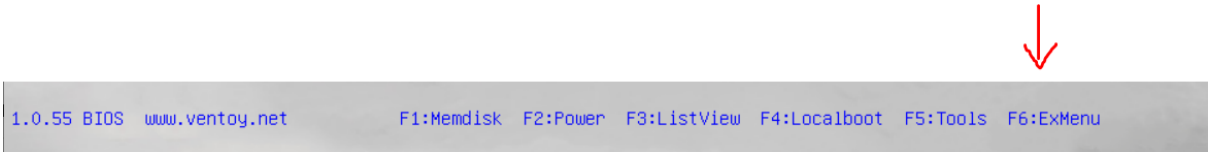
An example of our Ventoy.json with two menu_tip entries is shown on the page below (later versions of Ventoy 1.0.61+ can also display tips for directories too):

```
{
      "control": [{
                  "VTOY_DEFAULT_MENU_MODE": "1"
            },
            {
                  "VTOY_FILT_DOT_UNDERSCORE_FILE": "1"
            },
            {
                  "VTOY_TREE_VIEW_MENU_STYLE": "0"
            },
            {
                  "VTOY_DEFAULT_SEARCH_ROOT": "/ISO"
            },
            {
                  "VTOY_DEFAULT_KBD_LAYOUT": "QWERTY_UK"
            }
      ],
      "persistence": [{
            "image": "/ISO/LINUX/010 Ubuntu v20.04.3 (64-bit).iso",
            "backend": [
                  "/ISO/LINUX/ubuntu512ext4.dat",
                  "/ISO/LINUX/ubuntu512xfs.dat"
            ]
      }, {
            "image": "/ISO/LINUX/Ubuntu-v20.04.3.iso",
            "backend": "/ISO/LINUX/ubuntu512xfs.dat"
      }],

      "auto_install_legacy": [{
                  "image": "/ISO/WINDOWS
INSTALL/Win10_EnglishInternational_x64.iso",
                  "template": [
                        "/ISO/WINDOWS INSTALL/Win10ProLegacy.xml",
                        "/ISO/WINDOWS INSTALL/Win10ProLegacy2.xml"
                  ]
            },
            {
                  "image": "/ISO/WINDOWS
INSTALL/Win11_EnglishInternational_x64.iso",
                  "template": [
                        "/ISO/WINDOWS INSTALL/Windows 11 Home fix tpm.xml",
                        "/ISO/WINDOWS INSTALL/Windows 11 Pro fix tpm.xml"
                  ]
            }
      ],

      "auto_install_uefi": [{
                  "image": "/ISO/WINDOWS
INSTALL/Win10_EnglishInternational_x64.iso",
                  "template": [
                        "/ISO/WINDOWS INSTALL/Win10ProUEFI.xml",
                        "/ISO/WINDOWS INSTALL/Win10ProUEFI2.xml"
                  ]
            },
            {
                  "image": "/ISO/WINDOWS
INSTALL/Win11_EnglishInternational_x64.iso",
                  "template": [
                        "/ISO/WINDOWS INSTALL/Windows 11 Home fix tpm.xml",
                        "/ISO/WINDOWS INSTALL/Windows 11 Pro fix tpm.xml"
                  ]
```

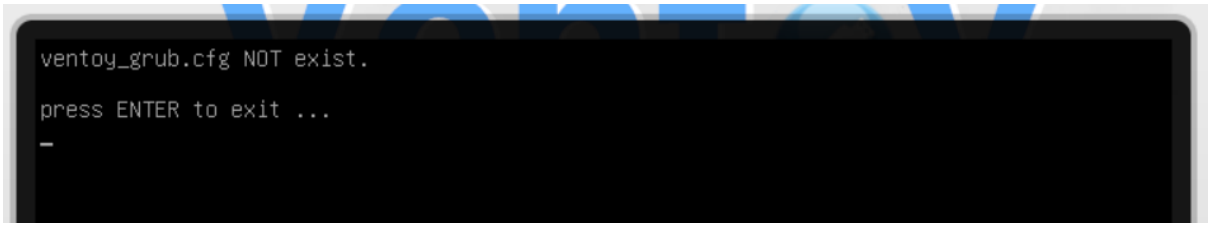
```
            }
      ],
      "image_blacklist_uefi": [
            "/ISO/ANTIVIRUS/comodo_rescue_disk_2.0.261647.1.iso",
            "/ISO/WINDOWS
INSTALL/windows_10_enterprise_ltsc_2019_x86_dvd.iso",
            "/ISO/WINPE/Win10PE_LegacyOnly_x86.iso"
      ],
      "menu_tip": {
            "left": "10%",
            "top": "81%",
            "color": "green",
            "tips": [{
      "image": "/ISO/WINDOWS INSTALL/Win11_EnglishInternational_x64.iso",
      "tip": "This ISO file contains Intel SSD/NVMe/Wifi/USB 3.0 drivers."
                  },
                  {
      "image": "/ISO/LINUX/010 Ubuntu v20.04.3 (64-bit).iso",
      "tip": "This ISO file can be run with Persistence"
                  }
            ]
      }
}
```

Chapter 24 - Add your own grub2 menus (Extension plugin)

You may have noticed the 'F6:ExMenu' text at the bottom of the Ventoy menu:



However, when you press F6 you will see the message below:



because a \ventoy\ventoy_grub.cfg file was not found on the first partition.

Below is a sample ventoy_grub.cfg menu (in grub2-speak) that was actually made for an Easy2Boot USB drive.

```
if [ -e ${vtoy_efi_part}/grubfmx64.efi ]; then

menuentry "Load agFM Menu" --class=custom {
if [ "${grub_platform}_${grub_cpu}" = "pc_i386" ]; then
    linux ${vtoy_efi_part}/e2b/loadfm
    initrd ${vtoy_efi_part}/e2b/grubfm.iso
else
    if [ "$grub_cpu" = "x86_64" ]; then chainloader
${vtoy_efi_part}/grubfmx64.efi; fi
    if [ "$grub_cpu" = "i386" ]; then chainloader
${vtoy_efi_part}/efi/boot/bootia32.efi; fi
fi
}

if [ "${grub_platform}_${grub_cpu}" = "pc_i386" ]; then
menuentry "Load Easy2Boot Menu" --class=custom {
      insmod ntldr
      set root=${vtoydev},1
      ntldr ${vtoy_iso_part}/grldr
}
fi
fi

menuentry '<-- Return to previous menu [Esc]' --class=vtoyret VTOY_RET {
    echo 'Return ...'
}
```

The first menu runs the grubfm iso only if it is in Legacy Mode, or runs the grubfmx64.efi boot file only if in UEFI64 mode or runs the bootia32.efi file if we booted on a UEFI system with a 32-bit CPU.

The second menu is only displayed in Legacy mode and boots to grub4dos (\grldr boot file).

This is an example of how you can write your own grub2 menu extension for Ventoy.

Later chapters give you a brief introduction to grub2 and you can find lots of sample grub2 menus which can boot from ISOs, etc. on the internet.

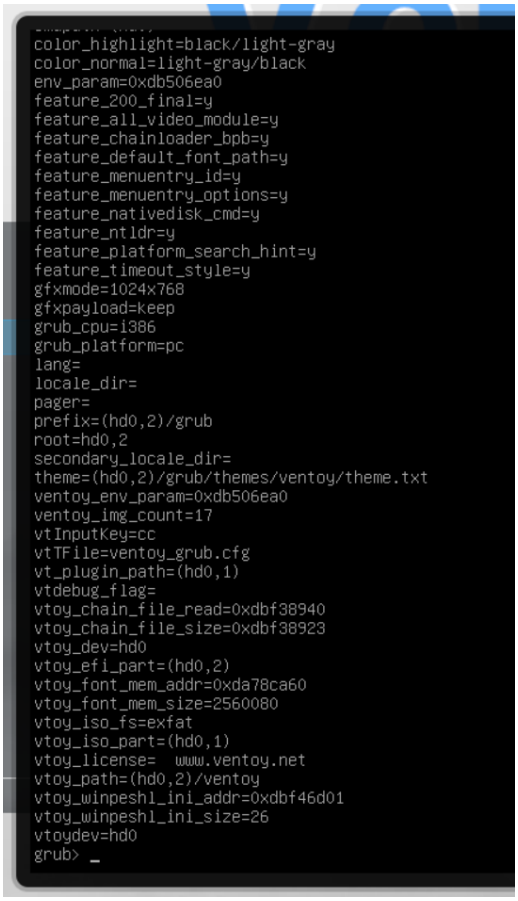
grub2 variables begin with a $ symbol and a few Ventoy grub2 variables are available for use within the Ventoy Extended menu system.

Name	Example
$vtoydev	hd1
$vtoy_iso_part	(hd1,1)
$vtoy_efi_part	(hd1,2)
$VENTOY_VERSION	1.0.64

Ventoy variable names are often enclosed by curly braces - e.g. `${vtoydev}`.

Note that when you Legacy boot, the USB drive is always hd0, however if you have UEFI-booted to Ventoy, the USB drive may not be hd0. That is why you need to use these variables instead of just using (hd0,1) for partition 1 and (hd0,2) for partition 2.

Tip: If you press 'c' in the Ventoy menu system (or your extended menu) to get to the grub2 command, you can then type 'set' to see all the local grub2 environment variables available - see the screenshot below:



Note that not all these variables may be available in your ventoy_grub.cfg menu.

Ventoy does support the booting if vhd files (even from a local disk) and you can use a menu such as:

```
menuentry "Boot My Linux VHD" {
    set my_vdisk_path="/VhdDir/Ubuntu.vhd.vtoy"

    if search -n -s vdiskhd -f "$my_vdisk_path"; then
        vtoyboot_common_func "($vdiskhd)$my_vdisk_path"
    else
        echo "$my_vdisk_path not found"
    fi
}
```

If you have a folder containing the BCD (which points to the VHD file) and a bootable UEFI Windows file on a different partition or disk, you can use grub2 to boot from a fixed-size VHD containing Windows:

```
if search --file /EFI/windows_vhd/bootmgfw.efi; then
      menuentry '[Find and start Microsoft Windows VHD system (UEFI mode)]'
--class vhd {
            search -n -s -f /EFI/windows_vhd/bootmgfw.efi
             chainloader / EFI/windows_vhd/bootmgfw.efi
      }
fi
```

It is also possible to use Ventoy to boot *some Linux ISOs* by using an unofficial menu entry such as the one shown below:

```
menuentry "Boot pmagic_2018_10_12.iso" {

    set vt_chosen_path="/_ISO/LINUX/pmagic_2018_10_12.iso"

    if [ -e  "${vtoy_iso_part}$vt_chosen_path" ]; then
          echo File found on ${vtoy_iso_part} ...
          ventoy_iso_busybox_ver
          if [ "$grub_platform" = "pc" ]; then
                  if vt_check_mode 0; then
                        legacy_iso_memdisk ${vtoy_iso_part}
"$vt_chosen_path"
                  else
                        legacy_iso_menu_func ${vtoy_iso_part}
"$vt_chosen_path"
                  fi
            else
                  if vt_check_mode 0; then
                        uefi_iso_memdisk ${vtoy_iso_part}
"$vt_chosen_path"
                  else
                        uefi_iso_menu_func ${vtoy_iso_part}
"$vt_chosen_path"
                  fi
            fi
    else
        echo "$vt_chosen_path not found"
    fi
}
```

This menu is a 'hack' and is NOT SUPPORTED and may not work with future versions of Ventoy. It does not work for Windows Install ISOs (does not mount the ISO) or on ISOs which are not on partition 1. It is shown for

your interest only! Note that the F1 (load to memory) state can cause the ISO to be loaded into memory if it was pressed first before you run this menu entry.

For instance, you could add multiple instances of this menu and then use the F6 key to show a menu containing your favourite Linux ISOs.

Here is an example cfg file for specific payload files and for both MBR and UEFI modes:

```
if ["$grub_platform" = "pc" ]; then
     menuentry'[Start the first hard disk of the host to start the boot
manager]' --class disk {
        set root=(hd0,1)
        chainloader + 1
        boot
    }

    menuentry'[Start the host's second hard drive to start the boot
manager]' --class disk {
        set root=(hd1,1)
        chainloader +1
        boot
    }

    menuentry'[Start the host's third hard drive to start the boot
manager]'- -class disk {
        set root=(hd2,1)
        chainloader +1
        boot
    }
   menuentry'[Find and start Microsoft Windows system]' --class nt6 {
        if search -n -s -f /bootmgr; then
            ntldr /bootmgr
        elif search -n -s -f /ntldr; then
            ntldr /ntldr
        else
            echo "Windows NOT found ..."
        fi
    }

    menuentry'[Find grub4dos Multifunctional Boot Manager]/grldr 'class
grub {
        if search -n- s -f /grldr; then
            ntldr /grldr
        else
            echo "Grub4dos NOT found ..."
        fi
    }

     menuentry "[Start GRUB2 File Manager GRUB2-based file
manager]/boot/grubfm.iso "--class grub2{
if search -n -s -f /boot/grubfm.iso; then
  linux /boot/loadfm
  initrd /boot/grubfm.iso
fi
    }

     menuentry "[Start MAXDOS System Maintenance Toolbox]/boot/MAXDOS.IMG
"--class dos {
if search -n -s -f /boot/MAXDOS.IMG; then
```

```
     linux16 ($root)/boot/
  memdisk initrd16 ($ root)/boot/MAXDOS.IMG
fi
    } menuentry

     "[boot bypassing boot password]/boot/KONBOOT.IMG "--class dos {
if search -n -s -f /boot/KONBOOT.IMG; then
     linux16 ($ root)/boot/
  memdisk initrd16 ($root)/boot/KONBOOT.IMG
fi
    } menuentry

     "[Start Clover Boot UEFI Loader (Legacy)] "--class clover{
if search -n -s -f /boot/Clover ; then
    linux /boot/loadfm
  initrd /boot/Clover
fi
    }


else


    menuentry'[Find and start Microsoft Windows system (UEFI mode)]' --
class nt6 {
        if search -n -s -f /EFI/Microsoft/Boot/bootmgfw.efi; then
            terminal_output console
            chainloader /EFI/Microsoft/Boot/bootmgfw .efi
            boot
        else
            echo "Windows NOT found ..."
        fi
    }

    menuentry'[Find boot system first boot boot manager (UEFI mode)]' --
class uefi {
        if search -n -s -f /efi/boot /bootx64.efi; then
            terminal_output console
            chainloader /efi/boot/bootx64.efi
            boot
        else
            echo "BOOTX64.EFI NOT found ..."
        fi
    }

   menuentry "[Start GRUB2 File Manager GRUB2-based file manager (UEFI
mode)]" --class GRUB2{
if search -n -s -f /boot/grubfmx64.efi; then
    chainloader /boot/grubfmx64.efi
fi
    }

  menuentry "[Start rEFInd multi-system boot manager utility]" --class
refind{
if search -n -s -f /refind/refindx64.efi; then
    chainloader /refind/refindx64.efi
fi
    } menuentry

     "[Start Clover EFI bootloader Clover EFI boot loader] "--class Clover{
if search -n -s -f /EFI/CLOVER/CLOVERX64.EFI; then
```

```
    chainloader /EFI/CLOVER/CLOVERX64.EFI
fi
    } menuentry

     "[Startup bypass Win&Mac boot password 】 "--Class kon{
if search -n -s -f /efi/boot/kon64.efi;then
    chainloader /efi/boot/kon64.efi
fi
    }

     menuentry "[Start Memtest86 Pro memory test] "--class Memtest{
if search -n -s -f /efi/boot/memtest64.efi; then
    chainloader /efi/boot/memtest64.efi
fi
    }    menuentry

     "[Lenovo One-key recovery UEFI version OKR 10.0.0.19 compatible
machine is available] "--class OKR{
if search -n -s -f /efi/boot/lenovookr.efi; then
    chainloader /efi/boot/lenovookr.efi
fi
    }

fi

menuentry '[Return to the previous menu] [Esc]' --class=vtoyret VTOY_RET
{echo'Return
    ...'
}
```

You can change each entry so that the menu entry will not be shown unless the payload file is actually found by using an 'if search --file' line, e.g.

```
if search --file /grldr ; then
    menuentry 'Load grub4dos multi-function boot manager /grldr' --class
grub {
        search -n -s -f /grldr
        ntldr /grldr
}
fi
```

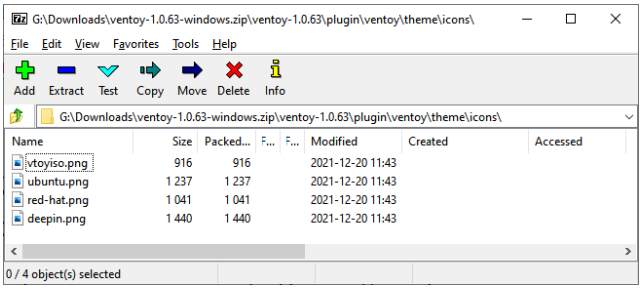
Note that if the same file (e.g. /grldr) exists on more than one partition within the system, this menu entry will only boot to the first payload file that is found by grub2.

You can use an 'if file exists' syntax to test for a file on a specific partition:

```
if [ -e ${(hd0,1)/grldr ]; then
    menuentry 'Load grub4dos multi-function boot manager /grldr' --class
grub {
        ntldr (hd0,1)/grldr
}
fi
```

Note that when you MBR\Legacy boot from a MBR Ventoy USB drive, the Ventoy USB boot drive is always hd0 and the first Legacy partition is number 1; however when you UEFI-boot from the USB drive, the boot drive can be any device number (e.g. hd3), so you should use the grub2 variable set up by Ventoy which is '${vtoy_iso_part}' instead of (hd0,1) for the first partition on the Ventoy USB drive.

The --class grub text string defines the name of the icon file which is located within the current grub2 theme. The default Ventoy theme only has four icon files (vtoyiso.png, ubuntu.png, red-hat.png, deepin.png) but you can use your own theme and add many more icon .png files.



Exercise 26: Make a simple ventoy_grub Extended menu

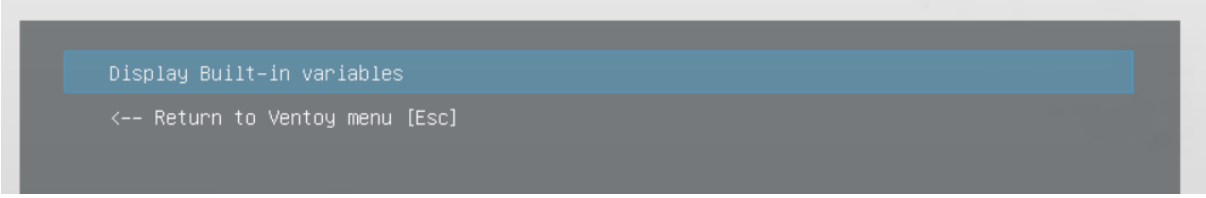
The creation and use of the extended menu is really only for use by grub2 experts but why not try it yourself anyway?

Here is an simple test menu to get you started:

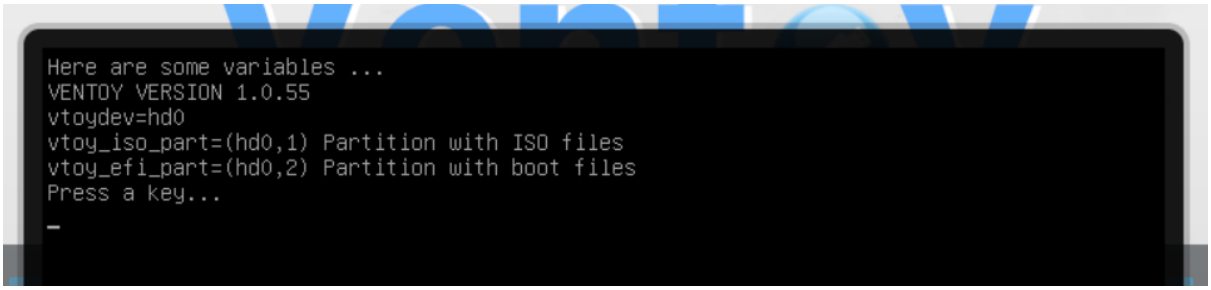
```
menuentry 'Display Built-in variables' --class=custom {
    echo "Here are some variables ... "
      echo VENTOY VERSION $VENTOY_VERSION
      echo vtoydev=${vtoydev}
      echo vtoy_iso_part=${vtoy_iso_part}     Partition with ISO files
      echo vtoy_efi_part=${vtoy_efi_part}     Partition with boot files
      echo Press a key...
      read
      clear
      echo Thank you
      sleep 1
}

menuentry '<-- Return to Ventoy menu [Esc]' --class=vtoyret VTOY_RET {
    echo 'Return ...'
}
```

1. Cut and paste the above menu into a \ventoy\ventoy_grub.cfg file.

2. Now boot to Ventoy and press F6...



and press ENTER to display the result...

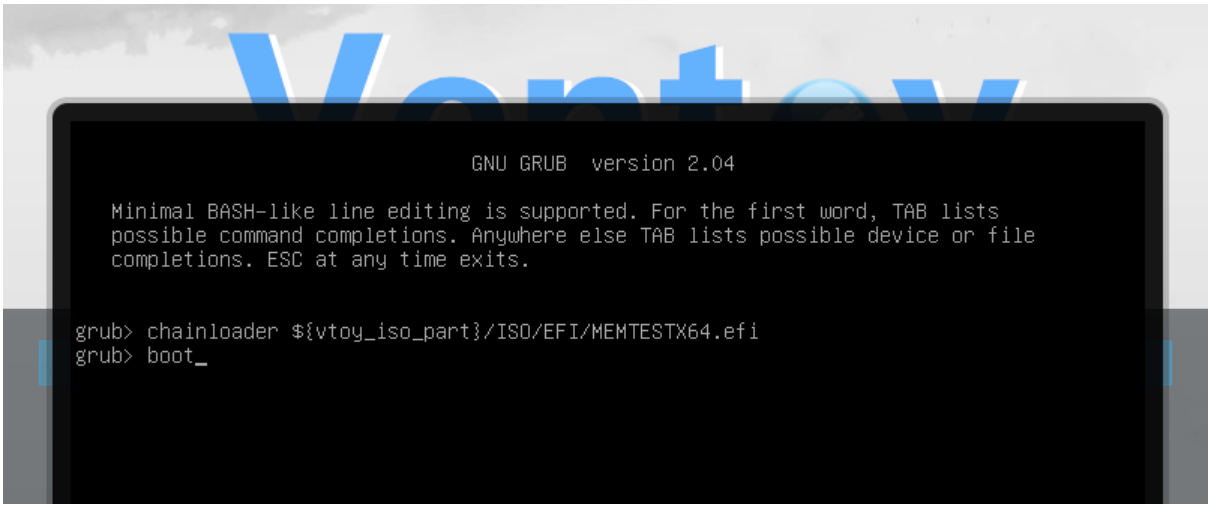


You can test out other commands by typing them in the grub2 command console (press 'c'). Type help for a list of commands and maybe see what happens if you type 'help reboot' and then just 'reboot'. ESC will return you to the Ventoy menu.

Now, as an exercise, try making two more menu entries in the ventoy_grub.cfg menu, one for rebooting the system and one for switching off the system (halt).

Tip: You can always test grub2 code directly on the grub2 command line.

In the case below, I have extracted the UEFI64 MemTest86 EFI boot file, renamed it and placed the file in the \ISO\EFI folder...



Once it works, the same commands can then be used in a ventoy_grub.cfg menu entry (the boot command is not needed in a menuentry):

```
menuentry 'Display Built-in variables' --class=custom {
    echo "Here are some variables ... "
      echo VENTOY VERSION $VENTOY_VERSION
      echo vtoydev=${vtoydev}
      echo vtoy_iso_part=${vtoy_iso_part}     Partition with ISO files
      echo vtoy_efi_part=${vtoy_efi_part}     Partition with boot files
      echo Press a key...
      read
      clear
      echo Thank you
      sleep 1
}

menuentry 'Boot MemTest86' --class=custom {
if [ "${grub_platform}_${grub_cpu}" = "pc_i386" ]; then
    echo
else
    if [ "$grub_cpu" = "x86_64" ]; then chainloader
${vtoy_iso_part}/ISO/EFI/MEMTESTX64.EFI; fi
    if [ "$grub_cpu" = "i386" ]; then chainloader
${vtoy_iso_part}/ISO/EFI/MEMTESTX86.EFI; fi
fi
}


menuentry '<-- Return to Ventoy menu [Esc]' --class=vtoyret VTOY_RET {
    echo 'Return ...'
}
```

Chapter 25 - Auto-select Memdisk mode for some payloads (Auto-memdisk)

After you press F1 in the Ventoy menu system, any ISO/IMG files that you have selected will be loaded into memory first before Ventoy loads the boot files within the ISO. This sometimes allows certain types of ISOs such as WinPE ISOs in Legacy mode to boot correctly.

The payload file should be under 1GB or so in size because RAM is consumed when a file is copied into memory.

For more details on the Memdisk mode, please refer MEMDISK.

With this plugin, you can set the memdisk mode for some ISO files and then Ventoy will always use Memdisk mode to boot them.

An auto_memdisk array is defined to describe the auto installation configuration in /ventoy/ventoy.json.

```
{
    "auto_memdisk": [
        "/ISO/mt531b.iso",
        "/ISO/ESXi.iso",
        "/ISO/Win10PE.iso"
    ]
}
```

Key	Type	Description
auto_memdisk	STRING	The full path of the iso file. Only ISO files are supported. This option supports fuzzy matching, see Path Matching.

Chapter 26 - Add passwords (Password plugin)

In the Ventoy.json file, you can set a password for the whole Ventoy menu and/or for individual files.

Key	Type	Description
bootpwd	STRING	Password when Ventoy is booting. Ventoy will exit after 3 retries.
isopwd	STRING	Default password for all .iso files. Will return to the main menu if input password is wrong.
wimpwd	STRING	Default password for all .wim files. Will return to the main menu if input password is wrong.
imgpwd	STRING	Default password for all .img files. Will return to the main menu if input password is wrong.
vhdpwd	STRING	Default password for all .vhd/.vhdx files. Will return to the main menu if input password is wrc
efipwd	STRING	Default password for all .efi files. Will return to the main menu if input password is wrong.
vtoypwd	STRING	Default password for all .vtoy files. Will return to the main menu if input password is wrong.
file	STRING	The full path of the image file. This option supports fuzzy matching, see Path Matching
parent	STRING	The full path of the parent directory. All files (.iso/.wim/.img ...) will use the same password. **Must not end in /**
pwd	STRING	Password when booting the image file. Will return to the main menu if input password is wror

For more details see the Password page.

Here is an example:

```
{
    "password": {
        "bootpwd": "txt#123456",
        "isopwd": "txt#111",
        "wimpwd": "txt#222",
        "imgpwd": "txt#333",
        "vhdpwd": "txt#444",
        "efipwd": "txt#555",
        "vtoypwd": "txt#666",
        "menupwd": [
            {
                "parent":"/MyVhdDir/WindowsVHD",
                "pwd":"md5#def#14fa11b4ab450b0204182266140e284d"
            },
            {
                "file":"/ISO/MX-19.*_x64.iso",
                "pwd":"txt#secret"
            },
            {
                "file":"/ISO/ubuntu-20.04-desktop-amd64.iso",
                "pwd":"md5#abc#0659c7992e268962384eb17fafe88364"
```

```
                }
            ]
        }
}
```

Note that when using a 'file' key, each * character denotes any *single character*. So in the menu above, any MX-19 ISO file version (MX-19.0 to MX19.9) which matches the pattern will prompt for the password ('secret').

Chapter 27 - Boot payloads from other partitions or disks

Ventoy v1.0.67 changed the F2 key function into a new disk 'Browser' function. This now allows us to browse to any disk partition within the system, select an .iso, .img. ,vhd, .wim. .vhdx, .vtoy or .efi file and boot from it (in much the same way that grubfm/agFM does).

Using plugins with payload files

If however, you want to use a plugin with the payload file and add this into ventoy.json, you must create .vlnk files and place these special files on Partition 1 of the Ventoy USB drive.

For instance, if you wanted to boot LinuxMint 20.3 ISO from a different disk or from USB partition 3 AND use a **persistence file**, you cannot specify the partition or disk within the ventoy.json file. This is because a windows volume letter of M: or a Linux volume name of \dev\sda2 will depend on what system you happen to be booting from and what disks are in that system. The ventoy.json file does not accept drive letters or Linux partition descriptions.

Ventoy solves this problem by using vlnk files (similar to shortcuts or symlinks). We can create these vlnk files using a utility provided by Ventoy and then place the vlnk files on the *main **Ventoy** Partition 1* volume. We can then specify these vlnk files in the ventoy.json configuration file because they are on the main payload partition.

Note that if the persistence file (or XML, kickstart or injection file) is on the main Ventoy Partition 1 then you don't need to use a vlnk file. If you want to specify XML or other plugin files, they must be located on Partition 1.

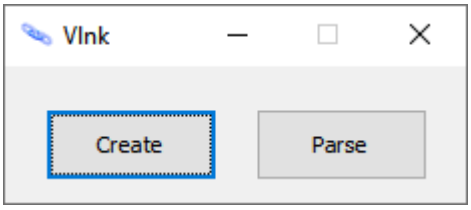
Exercise 27: Boot a Linux iso+persistence from any partition

1. Follow one of the previous exercises for a Linux ISO with persistence and ensure that the .iso file and the .dat file, which are both on the Ventoy USB disk, work correctly and as expected.

2. Copy both the .iso file and the .dat file to a different partition or disk. For instance, if your Ventoy USB drive has a Partition 3, you can copy the two files to the root folder or any folder on that Partition 3. If you want the files to be on a hard disk within your system then copy the two files there.

Note: You cannot have a .vlnk.iso file + a **.dat** file on Partition 1 (the persistence .dat file will be ignored) - you must use a **.vlnk.dat** file and it must point to the .dat file which is on the **same partition as the iso file**.

3. You now need to run VentoyVlnk (.exe or .sh) to create the vlnk files. Here is the Windows VentoyVlnk.exe app. which can be found in the ventoy-1.0.xx-windows.zip download file:



Just click on the 'Create' button and select the .iso file on whatever partition you have placed it (e.g. USB Ventoy Partition 3). A new 32Kb file of the same name but with a .vlnk.iso file extension will be created by the utility.

Repeat the 'Create' process but this time choose the .dat persistence file - a .vlnk.dat file will be created.

Note that the new vlnk files will be created in the same folder where the VentoyVlnk utility is located (not in the same partition as the target file).

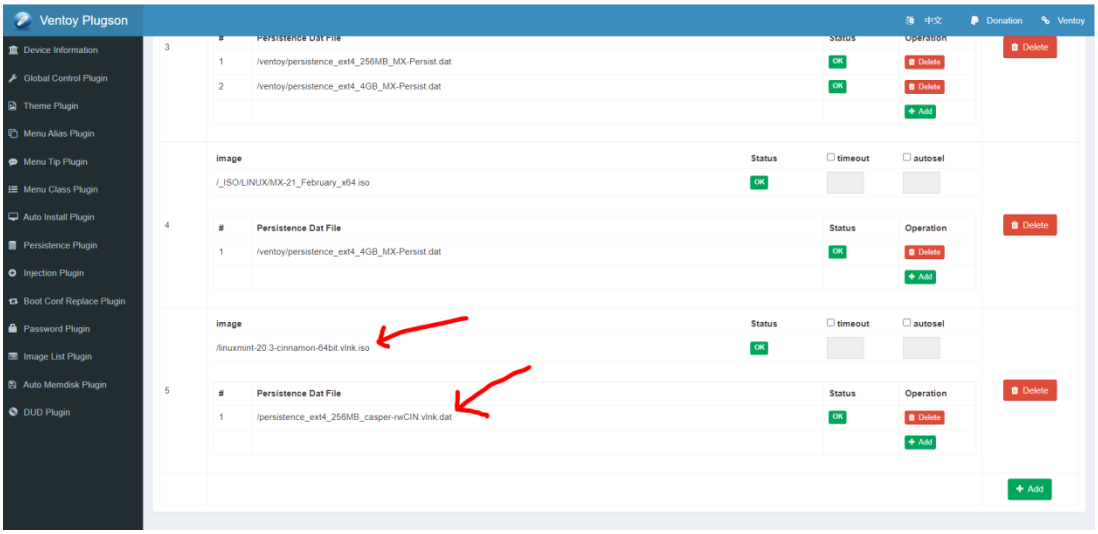
Note: The 'Parse' button can be used to select a vlnk file and it will tell you what the vlnk file points to.

4. Move the new .vlnk.iso and vlnk.dat files to a folder on the Ventoy USB Partition 1 (the main Ventoy payload partition).

Name	Date modified	Type	Size	Date created
persistence_ext4_256MB_casper-rwCIN.vlnk.dat	11/03/2022 11:49	DAT File	32 KB	11/03/2022 11:49
linuxmint-20.3-cinnamon-64bit.vlnk.iso	11/03/2022 11:49	ISO File	32 KB	11/03/2022 11:49

It is suggested that you create a new folder called perhaps 'LocalDisk' and copy all vlnk.* files into a folder under there...

5. Now use the VentoyPlugson app to modify the \ventoy\ventoy.json file to use these two new vlnk files (I just put the vlnk files in the root in this example)...



example \ventoy\ventoy.json section:

```
{
    "image": "/linuxmint-20.3-cinnamon-64bit.vlnk.iso",
    "backend":[
        "/persistence_ext4_256MB_casper-rwCIN.vlnk.dat"
    ]
}
```

6. You should now be able to boot to Ventoy and select the vlnk.iso file from the main Ventoy partition.

Note that if you use F2 and browse to the .iso file on Partition 3, it will simply boot the ISO without persistence.

Chapter 28 - Other Ventoy features

Not all of the features and functions of Ventoy have been covered in this introductory eBook.

Here are a few advanced features which you might find interesting:

- Linux vDisk plugin
- Linux .cfg bootfile replacement plugin
- Linux RHEL, CentOS Fedora, SUSE Driver Update plugin
- AutoInstallation script plugins (Kickstart.cfg, autoYast.xml,preseed.cfg) for Linux distros
- Linux and Windows file injection plugin
- Windows AutoRun batch file injection plugin - use with the injection plugin. There is a Turkish YT video here.

More features are being added to Ventoy every week, so I suggest you study the website documentation (topics in left-hand margin) to discover what else you can do with Ventoy!

Chapter 29 - How to recompile the Ventoy source code

Ventoy is open source and the code can be downloaded from GitHub.

If you wish to compile Ventoy, you can follow the instructions found in the BuildVentoyFromSource.txt file and in this article.

The Ventoy boot manager code is specifically written to run from a USB drive that has been prepared in a certain specific way. Ventoy expects the USB drive to have a 2nd partition with a volume name of VTOYEFI and it must be exactly 32MiB is size. On a Legacy disk, the 2nd partition must immediately follow the end of the first partition and there must be no gap between the two partitions - also the first partition must start at sector 2048. Other checks are also made for a GPT Ventoy disk.

You can reformat the first partition of a Ventoy disk but you should not resize it as this will cause the partitions to be moved and Ventoy will fail to boot. The 2nd partition must not be reformatted or changed in any way or else it may fail to be recognised when an update is performed.

'Ventoy for Easy2Boot' is a specially modified version of Ventoy which has had certain checks removed so that Ventoy will work on an Easy2Boot USB drive which contains a larger FAT Partition 2. Instructions for modifying the Ventoy source code to avoid these checks can be found in a chapter at the end of Easy2Boot eBook #4 - UEFI-multiboot using the a1ive grub2 File Manager.

You should not any report problems found with 'Ventoy for Easy2Boot' or your own altered compilation of Ventoy unless you can also reproduce the same problem on a true 'official' Ventoy USB disk.

Chapter 30 - Add Easy2Boot (Legacy) to Ventoy

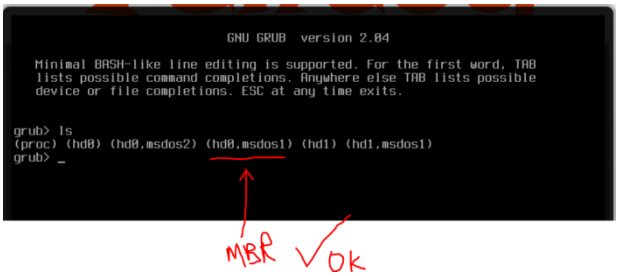
The **Easy2Boot** (E2B) multiboot USB menu system has been available for many years. It started as a grub4dos legacy-only based menu system, but then a grub2-based Legacy\UEFI menu system (agFM/grubfm) was added followed by the addition of a slightly modified version of Ventoy ('Ventoy for Easy2Boot').

You can easily add the *legacy-only* E2B boot files onto your *MBR-partitioned* Ventoy USB drive (it is not compatible with GPT-partitioned drives). From the Ventoy legacy menu you can then load the E2B legacy menu system. This will add the following E2B features:

1. Ability to install Windows 98/Me/XP from Install ISO files.
2. XP installs can automatically add the correct Mass Storage driver which means you can install XP directly from the ISO file without needing to first integrate the correct disk drivers for SCSI/SATA/RAID into the ISO file.
3. Ability to automate the install of Windows 7/8/10/11 from ISO, auto-install correct drivers, auto-install apps (online or offline) by using configuration file sets and XML files (SDI_CHOCO).
4. Design your own E2B menu/wallpaper/animation (E2B_Editor.exe tool included)
5. Boot older ISOs such as Hirens Boot CD 15, etc.
6. Boot to DOS/FreeDOS ISOs and .IMA files.
7. FreeDOS.ima included with NTFS driver (just copy your DOS executables to a folder on the USB drive, boot to FreeDOS and directly execute the DOS executable). Partition 1 must be FAT32 or NTFS for this.
8. Similar function to Ventoy for Windows (inc. XML files), WinPE, Linux ISOs inc. persistence.
9. May be able to boot to some 'difficult' Linux ISOs when Ventoy fails (ISO files must be contiguous).

Note that because the USB drive is not a 'E2B' drive, some features such as using Partition Images (.imgPTN) files to 'switch-in' complete partitions (e.g. Mac OSX or a full Linux/Windows OS) are not supported (but if you make and E2B USB drive and add 'Ventoy for Easy2Boot' to it, all features of E2B and Ventoy and agFM are supported). UEFI booting to agFM is also not possible.

Before you begin, check that your Ventoy USB drive has *standard BIOS MBR partitions*. An easy way to do this is to boot to the Ventoy menu - press 'c' - and type 'ls'. If **(hd0,msdos1)** is displayed then you have the correct partitions (if **(hd0,gpt1)** is displayed, then you have GPT partitions and E2B won't work correctly!).



I recommend that the main Ventoy partition is an NTFS volume - so after making a new Ventoy disk, just reformat the exFAT Partition 1 to NTFS, however the E2B menu also works if it is left as the default format of exFAT. By having an NTFS Partition 1, if you boot to the E2B FreeDos disk image in Easy2Boot, the DOS NTFS driver will allow you to access and run DOS executables on Partition 1 under FreeDOS.

E2B also requires that the last MBR Primary partition 4 slot is unused in the MBR on the Ventoy drive.

1. Download one of the E2B ZIP files (either the vanilla .zip file, or get the one with DPMS if you intend to add Windows XP ISOs so you can install that 20 year old OS).

Note: If you use E2B v2.11 or later then it will have the two files mentioned below in steps 3 and 4 already in it - see also the _ISO\docs\Sample mnu files\Ventoy folder.

2. Extract all the E2B files to the root of your main Ventoy Partition 1. You should end up with a dozen or so extra new files in the root and a new _ISO folder.

You can delete **\Make_E2B.exe** and the 500MB **_ISO\contig.iso** file to save space.

3. Make a new file at **_ISO\MAINMENU\ReloadVentoy.mnu** with this contents:

```
iftitle [vol (hd0,1) > (md)0x300+1 > nul ;; cat --locate=VTOYEFI (md)0x300+1] Load Ventoy (Legacy)
chainloader (hd0)+1
boot
```

4. Make a new file at **\ventoy\ventoy_grub.cfg** with this contents:

```
if [ -f ${vtoy_iso_part}/grldr -a -f ${vtoy_iso_part}/_ISO/e2b/grub/e2b.cfg -a "${grub_platform}_${grub_cpu}" =
"pc_i386" ]; then
menuentry 'Easy2Boot (MBR)' --class=custom {
        ntldr ${vtoy_iso_part}/grldr
        clear
}
fi

menuentry '<-- Return to Ventoy menu [Esc]' --class=vtoyret VTOY_RET {
  echo 'Return ...'
}
```

If you have already made your own ventoy_grub.cfg file, then just add the first menu (6 lines) to your existing file.

You can download the two files here (in E2B v2.11 the ReloadVentoy.mnu file and the ventoy_grub.cfg can be copied from the _ISO\docs\Sample mnu files\ventoy folder).

5. Now *legacy-boot* to Ventoy and then press **F6** to boot into Easy2Boot in legacy mode.

Payload files for use in the E2B menu system must be added to the _ISO\XXXX menu folders as with any Easy2Boot USB drive - see the website for details). Windows ISOs must go in the correct folder under _ISO\WINDOWS (e.g. _ISO\WINDOWS\WIN11).

Note that Partition Image (*.imgPTN*) files are NOT supported in E2B.

In the Easy2Boot menu, there should be a new menu entry to '**Load Ventoy (Legacy)**' which will return you back to the Ventoy menu system.

If you have any problems legacy-booting a payload using Ventoy, copy the file to the appropriate E2B menu folder and try the E2B menu system.

Some eBooks on Easy2Boot are also available ;-)

Chapter 31 - Important BIOS bugs and features that you really need to know about!

If you want your USB drive to be able to boot *from as wide a variety of systems as possible*, you should be aware of these bugs, because they are bound to catch you out sooner or later!

1. **The 128GiB MBR BIOS bug** - Some BIOSes have a buggy USB driver and they *cannot access sectors beyond 128GiB (137GB) on a USB drive*. The bug only affects MBR\CSM booting. It means that you should keep all MBR-bootable files within the first 128GiB of the start of the USB drive, otherwise you will get errors when accessing some outlying files. The easiest way to avoid this bug is to ensure that all partitions where you keep payload files are within 128GiB.

2. **'Floppy Disk' MBR BIOS bug** - Some BIOSes will assume that if a USB Flash drive has only one partition, it must be a USB Super-Floppy drive and will attempt to boot it as if it were a Super-floppy drive rather than a 'hard disk'. Ventoy drives contain at least two partitions.

3. **UEFI 'Out-of-order' MBR partition bug** - The MBR partition table contains up to four partition entries. Each partition table entry contains the Start Sector address of the partition. Some UEFI BIOSes require these Start Sector entries to be in ascending order otherwise *the user will not be offered a 'USB UEFI' boot option* and you can only CSM\Legacy boot.

4. '**UEFI-only' bug** - If the USB drive contains a FAT32 partition containing a \EFI\BOOT\BOOTx64.EFI file (assuming it is a UEFI64 BIOS), the BIOS will never *offer the user* a CSM\Legacy boot menu choice (even if you have enabled CSM support in the BIOS). This bug means that you should never have a \EFI folder or boot file present on a FAT partition if you want it to always CSM\Legacy boot. If you encounter this issue, you can temporarily rename the \EFI folder on Partition 2 so that you can CSM\Legacy boot.

5. **UEFI NTFS boot feature** - All UEFI BIOSes should be able to UEFI-boot from the default .EFI boot files on FAT partitions (FAT12/FAT16/FAT32), but may not boot from the same file if it is on an NTFS or exFAT partition. However, some UEFI BIOSes also contain an integrated UEFI NTFS driver. These UEFI BIOSes can boot from NTFS partitions as well as FAT partitions. This means that some systems (e.g. Asus) may be able to boot from a .EFI boot file on NTFS partition of a USB drive, whereas other systems will not UEFI-boot from the same drive.

6. **UEFI Secure Boot feature** - All UEFI BIOSes should have a Secure Boot option. *They will only boot from signed boot files* (and those boot files should only load signed kernels, etc.). This ensures that the boot files have not been infected/changed. All Windows OEM systems shipped by manufacturers should have Secure Boot enabled in the BIOS and a PK loaded. There is normally a boot setting to UEFI-boot from only Microsoft-signed files or 'Other' files.

7. **Non-standard FAT32 cluster sizes** - Most FAT32 partitions which are less than 32GB in size have a default cluster size of up to 16K, however FAT32 partitions larger than 32GB are not officially supported by Microsoft. Consequently some UEFI BIOSes do not understand FAT32 file systems with 32k or larger clusters and so will not boot from an EFI boot file on FAT32 partitions if they are larger than 32GB.

8. Secure UEFI BIOS is in 'User Platform Mode' - If Secure Boot is enabled and a PK and KEK enrolled, the pop-up BIOS Boot Selection Menu may only allow you to UEFI-boot from devices containing a matching, signed boot file (such as Microsoft, Ubuntu and MemTest86 EFI boot files). You may need to disable Secure Boot for some USB drives/EFI boot files.

9. **Overlapping partitions** - some UEFI BIOSes do not like overlapping partitions on a USB drive, for instance:

No.	Vol. Label	Drive	ID	Act.	Hid.	FileSys.	Start LBA	Total Sectors	Capacity
0	EASY2BOOT	L:	0C	A		FAT32	41821424	1597657	780.1 MB
1	E2B_PTN2	K:	0C			FAT32	209743116	40342260	19.2 GB
2	SWITCH_PE		07			NTFS	211746892	1039063	507.4 MB

The Start LBAs are all in ascending order, however the large second partition shown above includes a section of the disk that is in the third partition, so UEFI treats *all* of the disk partitions as invalid. This may cause there to be no UEFI boot option in the PC boot menu and no volumes will be displayed in the EFI command shell either. Also, Windows 10 only sees two partitions on such a USB drive and not three.

10. **GPT only** - some UEFI BIOSes seem to require the USB boot device to contain GPT partitions - they will not UEFI-boot from a FAT partition on an MBR partitioned USB drive. This contravenes the UEFI specification!

11. Windows and Linux - when these OS's write to a USB drive they can 'cache' the writes. What this means is that under Windows you can write sectors/files to a USB drive but the sectors on the drive are not actually written to until later (e.g. 30 seconds later). Under Linux this can be even longer.

- Under Linux always 'Eject/Unmount' the USB drive before removing it. You can also run the *sync* command from the console first.
- Under Windows, always use the 'Safely remove hardware and eject' taskbar control before removing a USB drive.
- If you are using a Windows virtual machine to boot from your USB drive, any changes you make to the USB drive may not take effect until 30 seconds or so later. For example, you can make a change to your USB drive and then immediately boot from the USB drive using the VM but you may then find that the change did not actually get written to the USB drive! This seems to occur with USB fixed-disks (removable\flash drives are usually written to immediately). Tip: If your USB drive has an activity LED then watch it carefully to see when data is accessed/written on the drive.

Chapter 32 - Bootable devices (USB 3 devices are best!)

Here are some choices:

- Flash drives - I recommend the SanDisk Extreme Pro USB 3.1 range of drives. Note that some other USB flash drives such as the Corsair GTX (very fast!) will appear to Windows as a 'Fixed Disk' and not as a 'Removable Disk'.
- USB Hard disks or USB HDD/SSD enclosures containing a hard disk or an inexpensive SSD.
- If using a USB hard disk, be sure it is NOT a native 4k sector hard disk. The drive must use 512-byte sectors to be BIOS compatible (512e mode).
- HDD enclosure emulators (e.g. IODD/Zalman) - these Hard Disk Drive USB enclosures can be used to load an ISO as a virtual USB CD\DVD device. You can then boot from the device just as if you were booting from a USB CD\DVD drive containing the CD\DVD - i.e. the device appears as a USB CD\DVD drive to the OS. These drives can also be used to boot to Ventoy and Easy2Boot as well, so you can have linux+persistence, WindowsToGo, etc. on the same drive. They are particularly useful for UEFI-booting because you can select any UEFI ISO and boot from it.
- Write-protected USB drives - Some USB drives (e.g. Netac U335, Kanguru, Corsair Padlock) - these have a switch for write-protection. You can write-protect the drive to avoid contamination from an infected OS. With some drives you can flip the WP switch as soon as the OS has booted.
- Hardware Encrypted USB drives (e.g. IODD 2541, IODD Mini SSD, Netac U618, DataShur) - these drives are also bootable but must be unlocked with a PIN code first before you can boot from them and there may be timing and device reset problems with some payloads (not recommended unless the USB drive contains confidential files).
- SD cards + USB card readers - I do not recommend using these because they can be quite slow unless you use a fast USB 3.0 card reader and a reasonably fast SD card. However, many include a write-protect tab which you may find useful. Note that many *internal card readers* on PCs and Notebooks do not support *booting* directly from SD cards. Oh, also did I mention this before, internal card readers tend to be *very slow*?

Note: Old Operating Systems, such as XP and Windows 7, do not 'understand' modern chipsets and modern devices such as **USB 3.0 ports** or **NVMe solid-state disks**, because they did not contain modern drivers.

IODD Mini

I also recommend the IODD Mini SSD which can directly MBR- and UEFI-boot from 99% of all ISOs and VHDs, etc.

It is solid state and has very few compatibility issues. I can be encrypted and so it is secure and can be write-protected when needed.

When used together with Ventoy or E2B (for automated Windows installs, Linux+persistence, AV+updates, etc.) it makes a perfect choice!

Chapter 33 - So how does UEFI-booting actually work then?

The old-style *MBR-booting* dates back at least to the 1980s and works by directly reading the bytes from *sectors* on a disk into memory and then executing those bytes as CPU operation codes (whatever they may be!).

However, UEFI BIOSes load boot *files* (not sectors). This means that they need to be able to understand the file system (e.g. FAT32) that is on the boot disk. UEFI BIOSes can also check if the file is 'signed' and if it is safe to boot from.

UEFI BIOSes follow this boot sequence (when in UEFI boot mode):

- Switch on
- Check non-volatile memory to obtain an ordered list of boot devices and boot files that has been previously defined by the user or set by a previous OS.
- Check for valid volumes (typically FAT12/FAT16/FAT32 or GPT EFI FAT volumes only).
- Look for each designated boot file in turn (default = \EFI\boot\bootxxxxx.efi where xxxxx depends on the CPU architecture - see below) - a pop-up Boot Selection Menu can also be used.
- Load the boot file into memory (if Secure Boot is enabled in the BIOS settings, the boot .EFI file signature must match one or more keys).
  Default FAT Boot file:
  o For a 64-bit x86 UEFI system, the default EFI boot file is \EFI\boot\bootx64.efi.
  o For a 32-bit x86 UEFI system, the default EFI boot file is \EFI\boot\bootia32.efi.
- Execute the code that was loaded into memory.

A UEFI system will either have 32-bit UEFI firmware or 64-bit UEFI firmware - it will not have both. It may also contain a 'CSM' code module which is a cut-down Legacy BIOS.

Some systems have UEFI firmware which can also access NTFS partitions, however this is not a standard requirement of UEFI specification - only FAT partitions and CD file systems will be 'universally' UEFI-bootable on all UEFI systems.

Note: The pop-up BIOS Boot Selection menu behaviour varies depending on the UEFI BIOS developer!

- Some BIOSes can access EFI boot files on both FAT and NTFS volumes
- Some BIOSes will add any partition containing a valid file system to the menu
- Some BIOSes will add partitions to the menu if they have valid \EFI\ folder
- Some BIOSes will add partitions to the menu if they have valid \EFI\Boot{arch}.EFI files
- Some BIOSes when in Secure Boot mode, will only list partitions that contain signed .EFI boot files.

Most UEFI-capable Operating Systems will program their device+path+boot file name into the UEFI BIOS NVRAM when they are installed - for instance, when installing Windows, it can specify that the UEFI firmware should load the \EFI\microsoft\boot\bootmgr.efi file from a particular hard disk and partition, instead of the default \EFI\boot\bootx64.efi file by telling the BIOS to add it into the NVRAM in the system. Thus if we have installed both Ubuntu and Windows 8 to the system, the UEFI may allow the user to choose to boot to either of these via the UEFI boot menu, because each OS has added a boot option into the UEFI non-volatile RAM storage area. In a typical UEFI setup menu, you can view these boot options and delete or clear the entries if you wish.

If no valid boot files are specified in the UEFI non-volatile RAM storage area or a different partition is selected by the user from a boot menu, then the default boot file will be used (e.g. \EFI\boot\bootx64.efi for an Intel x86 64-bit UEFI system or \EFI\boot\bootia32.efi for a UEFI32 BIOS).

Beware: Just because a system has a 64-bit CPU, does not mean it has 64-bit UEFI firmware! Some systems with 64-bit CPUs have 32-bit UEFI firmware and will only UEFI-boot to a 32-bit OS (they will look for the default boot file: \EFI\boot\bootia32.efi).

64-bit UEFI cannot boot from 32-bit UEFI boot files or Legacy\MBR boot files. 64-bit UEFI code is not compatible with 32-bit UEFI code or Legacy\MBR code because the CPU is placed into a different mode by the BIOS. You can however UEFI-boot (e.g. to grub2) and then boot to an MBR payload, as long as that payload kernel does not contain any code that uses the Legacy BIOS (many Legacy Linux kernels can be MBR-booted in this way).

Note: The file \bootmgr.efi is used by UEFI systems to boot from a CD/DVD (El Torito specification) - it is not used for USB/hard disk booting.

Windows boot chain:
BIOS (PCAT) : bootmgr {BootMgr stub -> contains embedded BootMgr.exe} -> \Boot\BCD -> WinLoad.**exe** -> Windows
64-bit (U)EFI : bootx64.efi or Bootmgfw.efi -> \EFI\Microsoft\Boot\BCD -> WinLoad.**efi** -> Windows
32-bit (U)EFI : bootia32.efi or Bootmgfw.efi -> \EFI\Microsoft\Boot\BCD -> WinLoad.**efi** -> Windows

Typical Linux boot chain:
BIOS (PCAT) : grub2/syslinux - kernel file - drivers, etc.
64-bit (U)EFI : grub2/syslinux - kernel EFI file - drivers, etc.

UEFI booting

For PCs containing an Intel architecture x86 CPU, there are two types of UEFI BIOS and thus three types of UEFI systems:

1. UEFI 64-bit BIOS with 64-bit CPU
2. UEFI 32-bit BIOS with 64-bit CPU
3. UEFI 32-bit BIOS with 32-bit CPU

Types 2 and 3 are not common but they do exist, typically they are found in systems which have only a small amount of solid-state memory (disk and RAM) because 32-bit OS's require less space than a 64-bit version of the same OS.

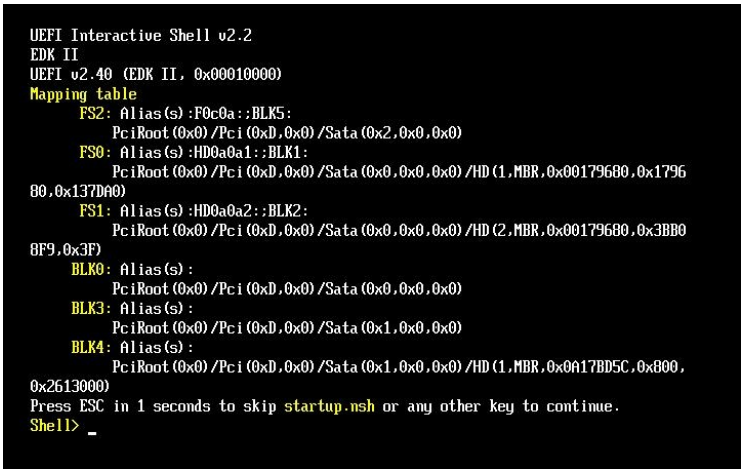
- A UEFI64 BIOS will only boot from a 64-bit EFI boot file (default file \EFI\BOOT\BOOTX64.EFI).
- A UEFI32 BIOS will only boot from a 32-bit EFI boot file (default file \EFI\BOOT\BOOTIA32.EFI).

Note: It is possible to UEFI64-boot to grub2 and run a 32-bit *Linux OS*.

The UEFI boot files should be on a FAT partition (FAT12/FAT16/FAT32) although some UEFI BIOSes can access NTFS partitions (they contain an NTFS driver), many cannot.

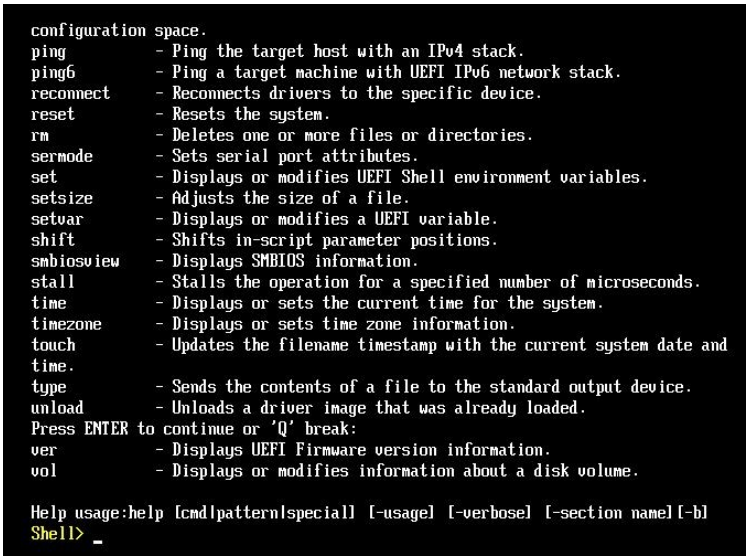
EFI Shell

When you UEFI boot, if no valid boot file can be found, then the EFI Shell may be loaded by the firmware instead.



Note that the default startup shell batch file called 'startup.nsh' will executed if present.

You can type **help -b** to view all available commands, one page at a time (using **-b**).



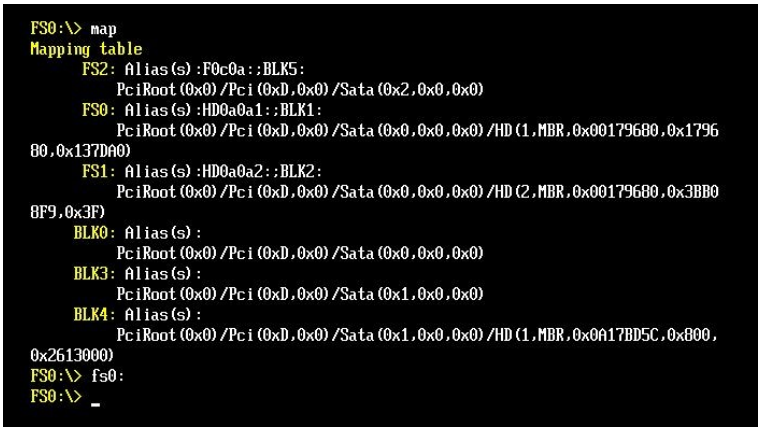
Typical switches are:

-b = paged output (may not be mentioned in help text!)
-v or **-verbose** = verbose output
-r = recursive
-q = quiet

You can use the TAB key for filename completion.

The first thing to do is to change the root...

Type map to see what file systems are available and then change the root by typing the FS name - e.g. **FS0:**



FS = filesystem (i.e. volume or partition), BLK = block device (e.g. hd0)

A command stack is normally available (hit cursor-up or cursor-down to go up or down the command list).

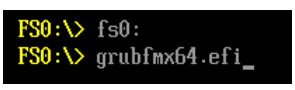
You can get help on any command - e.g. **help copy -b -v** or **copy -? -v -b**

Note that some commands may have more than one form (e.g. ls and dir, or cp and copy, del and rm).

Useful commands:

- **help -b** - list all help commands
- **help *(cmd)* -b -v** - show verbose help for *(cmd)*
- **bcfg boot dump** - list UEFI BIOS configuration boot list
- **dblk blk0** - dump first sector of blk0 drive to display
- **dblk blk0 C 2** - display data in LBA12 (sector 13) and LBA13 (sector 14)
- **devices** - list all detected devices
- **echo %fred%** - display value of variable named fred
- **edit \menu.lst** - edit a file
- **exit** - exit the shell
- **hexedit -f FS0:/menu.lst** - can edit a file, block device or memory region
- **ls -b -a -r FS0:\b*** - list files with attributes in all directories (can use wildcard * or ?)
- **map -v -b** - show information on mapped devices\filesystems
- **mode 128 40** - change screen size to 128 columns by 40 rows
- **reset** - reset the system
- **set fred 1** - set variable fred to 1 (use **set** to view all variables, **set fred** to display value, **set -d fred** to delete fred)
- **vol FS1:** - display volume info about FS1:
- filesystem commands: **attrib**, **ls**, **comp**, **cp**, **mkdir** , **mv**, **rm**, **touch**, **type** (**del**, **dir**, **copy**, **md**, **move**, **ren**)

Using the EFI shell, you can boot to the \grubfmx64.efi file to run agFM.



Chapter 34 - Secure Boot

UEFI Secure Boot requires that boot files are 'trusted'. They can be recognised as 'trusted' if they contain an embedded 'key' which has been signed by a certified 'authority' or if the user has added a certificate which

accepts a user 'key' which matches the same user key embedded within the boot file, or the user has added the 'hash' of that specific file into the UEFI BIOS settings as a 'trusted' file.

When UEFI-booting in Secure Boot mode, the first boot file should refuse to load the next boot file (e.g. kernel) unless that is also 'trusted' by the system ('platform'). This is a 'chain of trust' and ensures that the essential files in the 'boot chain' are 'pukka' and have not been tampered with.

About UEFI Security (PK, KEK, DB and DBX)

Devices (PCs) normally ship from an authorised Microsoft OEM with a vendor PK, Microsoft KEK and Microsoft DB keys. They may also include other keys.

Secure Boot certificates and hashes are recorded in the Trusted Platform Module (TPM) Platform Configuration Register (PCR) 7 for device integrity auditing purposes. The TPM hardware (chip) is included in every UEFI system and ensures the security of the whole Secure Boot platform. The UEFI BIOS can ask the TPM 'is this file trusted/signed/secure?'.

Platform Key (PK)

The PK functions as a top-level, root key for UEFI Secure Boot key storage. The PK is used to authorize runtime changes to Secure Boot keys and variables. The Platform Key establishes a trust relationship between the PC owner and firmware and is usually installed in the mainboards Non-Volatile Memory by the OEM PC manufacturer when the PC is built with Windows pre-installed.

If you have built your own system, a PK will not usually be loaded unless you have used the BIOS to load the 'defaults'. For instance, an ASUS motherboard could contain an 'ASUSTEK' PK.

Key Exchange Key (KEK)

Use the Microsoft KEK when booting a Microsoft OS. Key Exchange Key establishes trust between Operating Systems and the PC firmware. KEKs are installed into the PC by the OS and/or third party components which want to communicate with PC's firmware.

OEM PCs often ship with two KEKs installed, a Microsoft KEK and the manufacturers KEK so that it will accept the manufacturers UEFI update files.

Some devices allow an enterprise KEK *and* a Microsoft KEK. The enterprise KEK should sign DB and DBX keys used by the enterprise, organizations, or individual users as appropriate. The KEK can authorize runtime changes to some Secure Boot variables, DB keys and DBX keys. Note that the KEK does not need to be signed by a PK.

Most UEFI BIOSes will have an option to install the default KEKs.

Whitelist Database (DB)

The Authorized Database holding the public keys and certificates of any code module that is authorized to interact with PC firmware.

You can create one or more DB keys to be deployed by the enterprise, organizations, and/or users according to the device mission. Each DB key should be signed by the enterprise KEK. DB keys are used to sign trustworthy bootable drivers, binaries, and modules. SHA-256 hashes of trusted boot files may also be added into the DB. For instance, it may contain DB certificates for a Microsoft bootx64.efi file, a refind.efi file and a grubx64.efi file.

A boot file will be 'trusted' if it contains an embedded certificate key that is in the DB list, or if the boot file has a hash value which is in the DB list.

If you clear or load the default keys using the UEFI BIOS menu options, be sure to check that the 'state' of the DB (as well as KEK and PK) is set as 'Loaded', otherwise it will not even secure boot to a Microsoft Windows OS if Secure Boot is enabled!

Blacklist Database (DBX)

Place revoked, retired, or otherwise untrusted signing keys, KEKs and SHA-256 hashes of bootable content into the DBX to blacklist specific code modules.

Machine Owner Key (MOK)

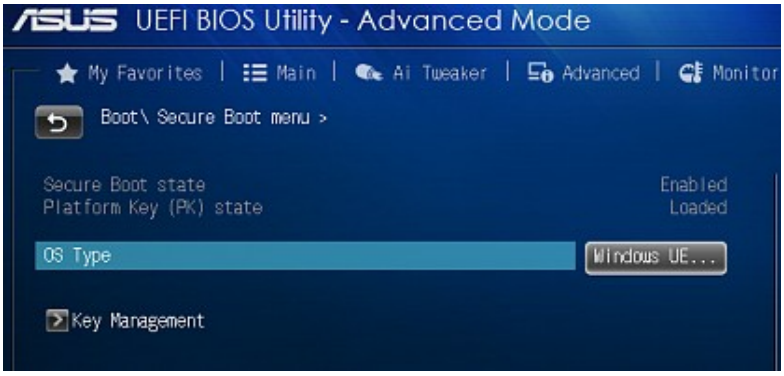
A MOK is a type of key that a user generates and uses it to sign an EFI binary. Any binary that contains that key will be accepted as secure. A MOK can also be a hash key generated from a specific file and added into the DB whitelist so that that single, specific file will be accepted.

How Secure Boot works

UEFI PC firmware will load the third party drivers, option ROMS and OS loaders that are signed by the Certification Authority (CA), in our case Microsoft. Any hardware vendor can write drivers for a UEFI BIOS and get it signed by Microsoft to run on UEFI PCs. OEMs install the public part of the key in the platform DB. The UEFI loader protocol service will validate the signature of the binary boot file against the authorized DB before it is allowed to run on the platform (e.g. a Microsoft signature in a bootmgfw.efi file). This chain of authentication can be continued from the UEFI to the OS loader and OS.

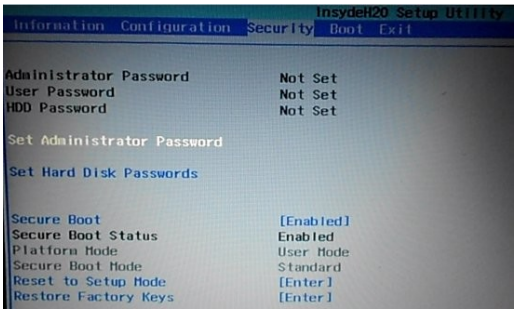
In summary, UEFI allows to run OS boot files that are signed and whose key is present in the DB (and not in the DBX list). This key mechanism ensures that OS boot files or option ROMs are allowed to execute only if they are authorized and have not been modified.

If Secure Boot is enabled in the BIOS, you will see an error if you attempt to boot to an unsigned, blacklisted or modified EFI boot file. If you boot to the rEFInd UEFI boot manager shim first, then because it is unsigned, the shim will fail to load the refind.efi file and instead it will load the Mok Manager UEFI application. The Mok Manager allows you to enrol a key (trust certificate) into the Non-Volatile RAM of the PC. This 'whitelists' the boot file by adding it to the DB list, so that from then on, the UEFI BIOS will 'trust' that particular boot file.



Platform Mode - When secure boot is first enabled, it is initially placed in **Setup Mode**, which allows a public key known as the Platform key (PK) to be written to the firmware. Once the key is written, secure boot enters **User Mode**, where only drivers and loaders signed with the platform key can be loaded by the firmware.

UEFI BIOSes that have a Platform Key (PK) set, may not allow you to boot from *any* unsigned .EFI boot file such as rEFInd or MokManager - no UEFI boot option will be displayed to the user in the BIOS Boot Menu.



It is therefore necessary to use the BIOS Setting option **Clear PK Key** (e.g. using the 'Reset to Setup Mode' which may have a description of "Clear PK, disable secure boot and enter Setup Mode."). Since this will remove security, you should insert a FAT32 flash drive and use the **Save PK Key** option to save the PK before clearing it.

If you do not clear the PK, and the BIOS is in Secure Mode, you will only be able to boot Microsoft-signed EFI boot files (and other whitelisted files) and you will not be able to boot to rEFInd or other unsigned boot files.

In the case of USB booting to rEFInd, an alternative is to use the BIOS configuration menu to load a certificate for rEFInd (see below).

If possible, I recommend temporarily disabling Secure Boot in the UEFI BIOS settings instead of using Mok Manager to enrol a hash key and compromising the security of the system. The Mok Manager can be unreliable (it may hang\crash) and it is also difficult to remove the enrolled hash key from some UEFI BIOSes once they have been added.

Secure Boot and Mok Manager

Some Ubuntu-based payloads and most Windows-based payloads contain signed EFI boot files. This means that we can directly Secure UEFI Boot from them. However, rEFInd and Ventoy grub2 is not signed and so we cannot directly Secure Boot to rEFInd unless it is added to DB and thus 'whitelisted'.

If we 'whitelist' rEFInd/grub2, when we Secure Boot we can then run the boot manager, however if we pick an EFI which has not been signed (e.g. KonBoot) then booting will still be denied.

Most Microsoft EFI boot files are signed and also some modern versions of the grub2 EFI (e.g. grubx64.efi) from Ubuntu or RedHat, as well as recent versions of MemTest86 from PassMark. These boot loaders will, in turn, only load EFI files that the BIOS regards as trusted.

If your system has UEFI Secure Boot enabled, it may prevent rEFInd or an unsigned grub2 from loading and you will not see any menu at all. In this case you will need to use MokManager which should load automatically, or use the UEFI BIOS configuration menu to add in a certificate for the efi binary.

MokManager.efi will help you enrol certificates (e.g. the \refind.cer certificate) or whitelist specific boot files if the BIOS refuses to run them. It hangs on some systems however :-(.

HashTool.efi will add/append the hash of a specific boot file to the UEFI BIOS whitelist. You can add KeyTool.efi to the whitelist using this tool. This tool is useful to whitelist boot files which will not Secure Boot.

KeyTool.efi will allow you to Edit and Save keys. You may need to add it to the whitelist (DB) using HashTool first before it will be allowed to run.

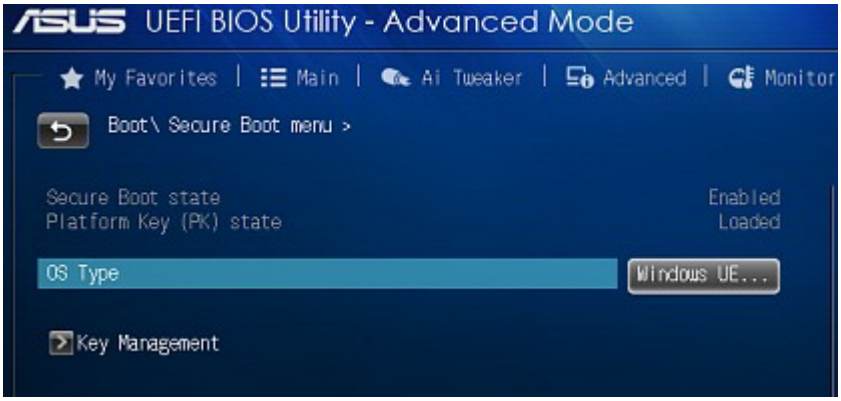
Tip: In case of any problems, you should use your BIOS configuration menu to **save all the keys to a USB drive** and then clear\reset all the BIOS configuration settings (try clearing the PK key and then try all Secure Boot keys first and if still in trouble, a generic BIOS 'Reset to Defaults' - make sure the DB whitelist is marked in the BIOS as 'loaded').

How to disable Secure Boot

Disabling Secure Boot (but still enabling UEFI-boot) will be very BIOS dependent.

- Go into the UEFI BIOS (Advanced) settings and find the Secure Boot option. If it is greyed out, you may first need to set a Supervisor or Master BIOS password first, and then reboot, enter the Supervisor password and go back into the UEFI Settings menu.
- Make sure the "OS Type" is "Windows UEFI" before disabling Secure Boot.
  Sometimes changing this option will disable Secure Boot.
  Sometimes adding CSM = Enabled will disable Secure Boot.
- Disable any Secure Boot option and reboot.

  Tip: You will usually need to **reboot back into the BIOS utility** to see that the 'Secure Boot State' has been changed.

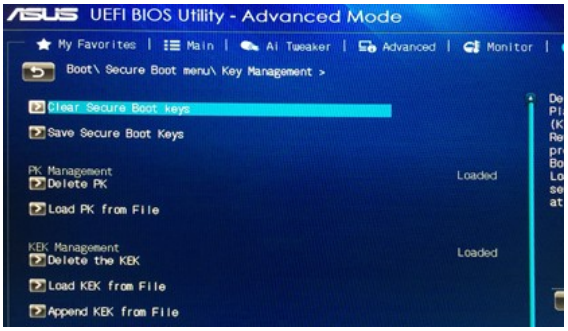


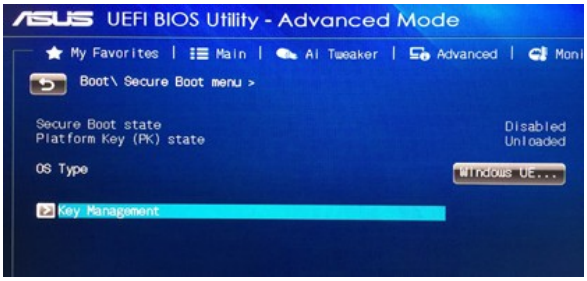
However, in some cases, a system may refuse to disable Secure Boot...

2. Check the PK State. If it is '**Loaded**' and Secure Boot is **enabled**, you may need to save the PK state to a USB FAT32 flash drive (e.g. 'Export Secure Boot Variables' or 'Save Secure Boot keys') and then Clear the PK using the **Key Management** options use 'Delete PK' if available. Take a backup copy of the saved key files for safety.

Note: It is often necessary to **reboot back into BIOS Setup** after making Secure Boot state changes, to allow you to change some of the other boot options.

3. Disable Secure Boot and set the 'OS Type' to 'Other'.





You will probably also want to **Enable CSM** and **Disable Fast Boot** so that it will MBR-boot from the USB drive as well as UEFI-boot.

3. To restore Secure Boot and load the original PK, you will need to find the '**Load PK**' menu option to load the key from your backup USB drive ('load defaults=No') and then load it as a '**UEFI Secure Variable'**.

Note: In some BIOSes, 'Load xx from File' and then choosing 'No' will load the default values.

Note: Some PC BIOSes have a bug and they will not present the user with any Legacy\MBR boot option. If you have one of these systems, then UEFI-boot to WinPE and temporarily rename the \EFI folder so that it is not seen by the UEFI BIOS when you need to boot to the MBR\Legacy menu.

Note that some UEFI BIOSes may refuse to provide a MBR-boot option and will only allow you to UEFI-boot. In this case you have some choices:

- Temporarily rename the \EFI folder on the FAT partition. You can UEFI-boot and then use WinPE to do this or use a Windows or Linux system. This will disable UEFI-booting from the drive until you restore the \EFI folder.

- Keep this drive just for UEFI-booting but make another drive without the FAT \EFI files for MBR-booting!

Chapter 35 - Grub2 configuration file syntax

grub cfg files are written in grub2's built-in scripting language, which has a syntax which is similar (but not *that* similar!) to that of GNU Bash and other Bourne shell derivatives.

Words

A word is a sequence of characters considered as a single unit by grub2. Words are separated by metacharacters, which are the following characters plus a *space*, *tab* or *newline*:

```
{ } | & $ ; < >
```

Note that the semi-colon symbol ; is typically used in place of a newline character.

Quoting may be used to include meta-characters in words - see below.

Reserved words

Reserved words have a special meaning to grub2. The following words are recognised as reserved when unquoted and either the first word of a simple command or the third word of a *for* command:

```
! [[ ]] { }
case do done elif else esac fi for function
if in menuentry select then time until while
```

Not all of these reserved words have a useful purpose yet; some are reserved for future expansion.

Quoting

Quoting is used to remove the special meaning of certain characters or words. It can be used to treat meta-characters as part of a word, to prevent reserved words from being recognised as such, and to prevent variable expansion.

There are three quoting mechanisms: the escape character, single quotes and double quotes.

A non-quoted backslash (\) is used as the **escape character**. It preserves the literal value of the next character that follows, with the exception of newline.

Enclosing characters in **single quotes** preserves the literal value of each character within the quotes. A single quote may not occur between single quotes, even when preceded by a backslash.

Enclosing characters in **double quotes** preserves the literal value of all characters within the quotes, *with the exception of '$' and* '\'. The '$' character retains its special meaning within double quotes. The backslash retains its special meaning only when followed by one of the following characters: '$', '"', '\', or newline. A backslash-newline pair is treated as a line continuation (that is, it is removed from the input stream and effectively ignored). A double quote may be quoted within double quotes by preceding it with a backslash (\").

Variable expansion

The '$' character introduces variable expansion. The variable name's value to be expanded may be enclosed in braces {}, which are optional but serve to protect the variable to be expanded from characters immediately following it which could be interpreted as part of the name.

Normal variable names begin with an alphabetic character, followed by zero or more alphanumeric characters. These names refer to entries in the grub2 environment (see Environment).

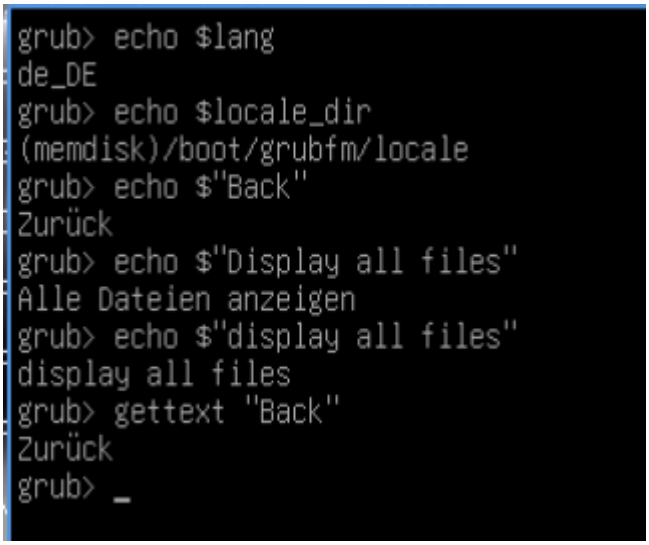
Positional variable names consist of one or more digits. They represent parameters which can be passed to function calls, with '$1' representing the first parameter, and so on.

The special variable name '**?**' expands to the exit status of the most recently executed command. When positional variable names are active, other special variable names '**@**', '*****' and '**#**' are also defined and these

three characters expand to: all positional parameters with necessary quoting, positional parameters without any quoting, and positional parameter count respectively.

Locale strings

grub2 can expand a string beginning with $" using a locale .mo file (e.g. locale/de_DE.mo). The .mo file will contain a list of all the exact English words or phrases and also each translated word or phrase. e.g. `echo $"Display all files"` will display 'Alle Dateien anzeigen' if that *exact* string and its translation are inside the .mo file. The variable `$locale_dir` specifies the folder containing the .mo file, and `$lang` specifies the .mo file name. If no exact match is found in the .mo file, then the string is not translated. If locale_dir is not set then translation is disabled.



Comments

A word or line beginning with '#' causes that word and all remaining characters on that line to be ignored.

Simple commands

A simple command is a sequence of words separated by spaces or tabs and terminated by a semicolon (;) or a newline. The first word specifies the command to be executed. The remaining words are passed as arguments to the invoked command.

The return value of a simple command is it's exit status. If the reserved word character ! precedes the command, then the return value is instead the logical negation of the command's exit status (read ! as 'NOT').

Compound commands

A compound command is one of the following:

`for` *`name`* `in` *`word`* `…; do` *`list`*`; done`

The list of words following *in* is expanded, generating a list of items. The variable name is set to each element of this list in turn, and list is executed each time. The return value is the exit status of the last command that executes. If the expansion of the items following in results in an empty list, no commands are executed, and the return status is 0.

`if` *`list`*`; then` *`list`*`; [elif` *`list`*`; then` *`list`*`;] … [else` *`list`*`;] fi`

The if list is executed. If its exit status is zero, the then list is executed. Otherwise, each elif list is executed in turn, and if its exit status is zero, the corresponding then list is executed and the command completes. Otherwise, the else list is executed, if present. The exit status is the exit status of the last command executed, or zero if no condition tested true.

```
while cond; do list; done
until cond; do list; done
```

The while command continuously executes the do list as long as the last command in `cond` returns an exit status of zero. The *until* command is identical to the *while* command, except that the test is negated; the do list is executed as long as the last command in `cond` returns a non-zero exit status. The exit status of the while and until commands is the exit status of the last do list command executed, or zero if none was executed.

```
function name { command; … }
```

This defines a function named name. The body of the function is the list of commands within braces, each of which must be terminated with a semicolon or a newline. This list of commands will be executed whenever name is specified as the name of a simple command. Function definitions do not affect the exit status in $?. When executed, the exit status of a function is the exit status of the last command executed in the body.

Menu entries can be specified using the menuentry word.

```
menuentry title [--class=class …] [--users=users] [--unrestricted] [--
hotkey=key] [--id=id] { command; … }
```

See menuentry for more details.

Built-in Commands

Some built-in commands are also provided by the grub2 script to help script writers perform actions that are otherwise not possible. For example, these include commands to jump out of a loop without fully completing it, etc.

```
break [n]
```

Exit from within a for, while, or until loop. If n is specified, break n levels. n must be greater than or equal to 1. If n is greater than the number of enclosing loops, all enclosing loops are exited. The return value is 0, unless n is not greater than or equal to 1.

```
continue [n]
```

Resume the next iteration of the enclosing for, while or until loop. If n is specified, resume at the nth enclosing loop. n must be greater than or equal to 1. If n is greater than the number of enclosing loops, the last enclosing loop (the top-level loop) is resumed. The return value is 0 unless n is not greater than or equal to 1.

```
return [n]
```

Causes a function to exit with the return value specified by n. If n is omitted, the return status is that of the last command executed in the function body. If used outside a function the return status is false.

```
setparams [arg] …
```

Replace positional parameters starting with $1 with arguments to setparams.

```
shift [n]
```

The positional parameters from n+1 … are renamed to $1…. Parameters represented by the numbers $# down to $#-n+1 are unset. n must be a non-negative number less than or equal to $#. If n is 0, no parameters are changed. If n is not given, it is assumed to be 1. If n is greater than $#, the positional parameters are not changed. The return status is greater than zero if n is greater than $# or less than zero; otherwise 0.

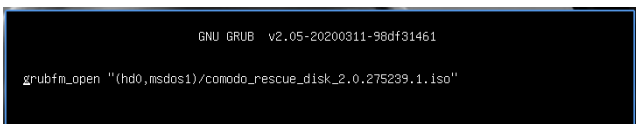
Grub2 standard environment variables

Grub2 standard commands

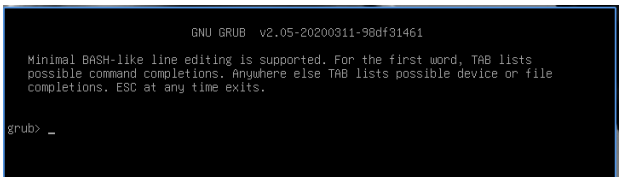
- integer1 -ge integer2 - integer1 is greater than or equal to integer2
- integer1 -gt integer2 - integer1 is greater than integer2
- integer1 -le integer2 - integer1 is less than or equal to integer2
- integer1 -lt integer2 - integer1 is less than integer2
- integer1 -ne integer2 - integer1 is not equal to integer2
- prefixinteger1 -pgt prefixinteger2 integer1 is greater than integer2 after stripping off common non-numeric prefix.
- prefixinteger1 -plt prefixinteger2 - integer1 is less than integer2 after stripping off common non-numeric prefix.
- file1 -nt file2 - file1 is newer than file2 (modification time). Optionally numeric bias may be directly appended to –nt in which case it is added to the first file modification time.
- file1 -ot file2 - file1 is older than file2 (modification time). Optionally numeric bias may be directly appended to –ot in which case it is added to the first file modification time.
- -d file - file exists and is a directory
- -e file - file exists
- -f file - file exists and is not a directory
- -s file - file exists and has a size greater than zero
- -n string - the length of string is nonzero
- string - string is equivalent to -n string
- -z string - the length of string is zero
- (expression) - expression is true
- ! expression - expression is false
- expression1 -a expression2 - both expression1 and expression2 are true
- expression1 expression2 - both expression1 and expression2 are true. This syntax is not POSIX-compliant and is not recommended.
- expression1 -o expression2 - either expression1 or expression2 is true

Chapter 36 - grub2 troubleshooting

If you press 'e' when a menu entry is highlighted, grub2 will display the code that will be executed in the menu entry...



The grub2 environment in the menu system also allows you to press 'c' at any time to drop to the grub2 command console.



You can type `help` and other commands in the console:

- `help` - list all grub2 commands
- `help chainloader` - show help for the chainloader command
- `set pager=1` - turn on paging (stops scrolling of screen)
- `set` - show all grub2 environment variable names
- `echo $vtoy_part` - displays the value of the variable *vtoy_part*
- `read` - waits for user input + ENTER key
- `sleep 3` - delays for 3 seconds
- `echo fred=*${fred}*` - displays the variable value with * on either side to show any hidden space characters within the value of the variable fred.

I use VirtualBox+VMUB to MBR or UEFI-boot to a USB drive and then test my grub2 scripts.

Tip: If you hold down the Ctrl key before clicking the Start button in VMUB, then you can edit and re-save the script file and re-test it without needing to exit VBox and restart it (but some payloads will not work correctly in this mode).

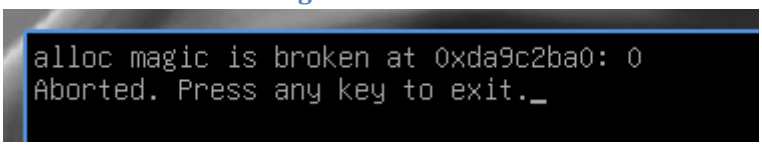
Linux ISO boot issues

The most common error when booting from Linux ISOs is that the second stage boot does not work and you end up with some sort of *initramfs* error because the kernel is trying to load a file which is actually inside the ISO file rather than on a CD\disk. When booting from an ISO, you either need to specify a very specific Linux kernel cheat code containing the exact path of the ISO file:
(e.g. iso-scan/filename=xxx)

If you have booted into the *Linux* console, you can also try these Linux commands...

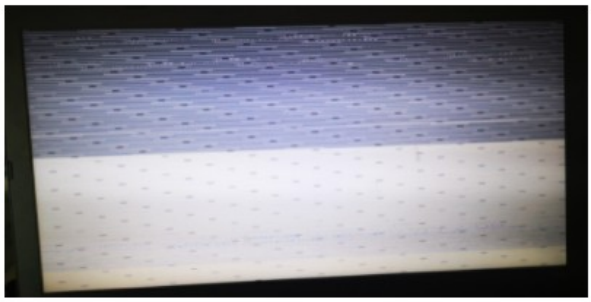
- `dmesg` - display all previous boot messages
- `df` - display devices
- `fdisk -l` - display disks
- `mount` - show mounted devices\volumes

Theme errors - 'alloc magic is broken'



Just about any type of syntax error in a *.txt themes* config file will cause this error with no clue as to what line caused it! Check for correct filenames, case, spelling and syntax errors.

Corrupt screen/bad screen resolution



When UEFI-booting, the OS that loads may switch the screen to an unsuitable graphics mode. It could be trying to set too high a resolution for the monitor or graphics adaptor.

Temporarily switch to text mode by pressing F7 in the Ventoy menu before selecting the ISO file.

If this works, you can set the Ventoy menu to use text mode by default:

Set the theme in the Ventoy.json file as follows:

```
{
    "theme": {
        "display_mode": "CLI"
    }
}
```

Alternatively, try adjusting the Ventoy menu graphics resolution before selecting the ISO file.

In the Ventoy' main menu, use F5 Tools --> Resolution Configuration to adjust the Resolution, and then boot to the ISO file. In most cases, choosing a smaller resolution will be better, for example 800x600x16 or 800x600x32.

Chapter 37 - Useful links

Ventoy - main site

Ventoy Forum - Official forum

Report a Ventoy bug - github Issue list

MajorGeeks - Download software.

FossHub - Download software.

FileHippo - Download software.

Sourceforge - Open Source software.

Live CDs list - List of useful CDs (outdated but still useful).

Distro Watch - Get the latest Linux distros.

SARDU's list if ISOs - SARDU is a USB multiboot solution.

grubfm - grub2-based multiboot project by a1ive.

RMPrepUSB website - RMPrepUSB.exe is a Windows utility for preparing/fixing bootable USB drives and over 100 useful articles

Easy2Boot - a multiboot USB drive which includes Ventoy and two other menu systems. Easy2Boot has better compatibility for Legacy booting and supports floppy boot images, DOS ISOs, XP ISOs and other power features such as switching in files as partitions to allow booting from HFS partitions to boot MAC install files or ext partitions to boot full Linux OS's, or partitions with Secure Boot code, etc. It also includes agFM (a modified version of grubfm) and 'Ventoy for Easy2Boot' (a slightly modified version of Ventoy).

Ventoy for Easy2Boot - slightly modified version of Ventoy adapted to work on an E2B USB drive.

JSON syntax checker - paste in text and click 'Validate JSON' button to check syntax.

JSON Pro syntax checker - paste in text and click 'Lint' button to check syntax.