rEFInd

Related articles

Arch boot process

Boot loader

Unified Extensible Firmware Interface

EFISTUB

booster

rEFInd (https://www.rodsbooks.com/refind/) is a UEFI boot manager capable of launching EFISTUB kernels. It is a fork of the no-longer-maintained rEFIt and fixes many issues with respect to non-Mac UEFI booting. It is designed to be platform-neutral and to simplify booting multiple operating systems.

Note: In the entire article *esp* denotes the mountpoint of the EFI system partition.

Contents

Installation

Install the **refind (https://archlinux.org/packages/?name=refind)** package.

Installing the rEFInd Boot Manager

rEFInd ships with **UEFI drivers** that implement **read-only** support for ReiserFS, Ext2, Ext4, Btrfs, ISO-9660 and HFS+. Additionally rEFInd can access any file system that UEFI itself can, that includes FAT (as mandated by the UEFI specification), HFS+ on Macs and ISO-9660 on some systems.

To find additional drivers see **The rEFInd Boot Manager: Using EFI Drivers: Finding Additional EFI Drivers (https://www.rodsbooks.com/refind/drivers.html#finding)**.

To use the rEFInd, you must install it to the EFI system partition either using the **refind-install script** or **by copying the files and setting up the boot entry manually**.

Warning: Your kernel and initramfs must reside on a file system that rEFInd can read.

Installation with refind-install script

The rEFInd package includes the *refind-install* script to simplify the process of setting rEFInd as your default EFI boot entry. The script has several options for handling differing setups and UEFI implementations. See **refind-install(8) (https://man.archlinux.org/man/refind-install.8)** or read the comments in the install script for explanations of the various installation options.

For many systems it should be sufficient to simply run:

```
# refind-install
```

This will attempt to find and mount your **ESP**, copy rEFInd files to *esp*/EFI/refind/ , and use **efibootmgr** to make rEFInd the default EFI boot application.

Alternatively you can install rEFInd to the default/fallback boot path *esp*/EFI/BOOT/bootx64.efi . This is helpful for bootable USB flash drives or on systems that have issues with the NVRAM changes made by *efibootmgr*:

```
# refind-install --usedefault /dev/sdXY
```

Where */dev/sdXY* is your EFI system partition (the block device, not its mountpoint).

Tip: By default refind-install installs only the driver for the file system on which kernel resides. Additional file systems need to be installed manually by copying them from /usr/share/refind/drivers_x64/ to *esp*/EFI/refind/drivers_x64/ , or you can install all drivers with the --alldrivers option. This is useful for bootable USB flash drives.

After installing rEFInd's files to the ESP, verify that rEFInd has created refind_linux.conf containing **kernel parameters** in the same directory as your kernel. Then this will not be created if you used the --usedefault option, run mkrlconf as root to create it. This configuration file will not be created if you used the --usedefault option, run mkrlconf as root to create it.

> **Warning:** When refind-install is run in chroot (e.g. in live system when installing Arch Linux) /boot/refind_linux.conf is populated with kernel options from the live system not the one on which it is installed. Edit /boot/refind_linux.conf and make sure the **kernel parameters** in it are correct for your system, otherwise you could get a kernel panic on your next boot. See **#refind_linux.conf** for an example file.

By default, rEFInd will scan all of your drives (that it has drivers for) and add a boot entry for each EFI bootloader it finds, which should include your kernel (since Arch enables **EFISTUB** by default). So you may have a bootable system at this point.

Secure Boot

See **Managing Secure Boot (https://www.rodsbooks.com/refind/secureboot.html)** for **Secure Boot** support in rEFInd.

Using PreLoader

See **Secure Boot#Set up PreLoader** to acquire signed PreLoader.efi and HashTool.efi binaries.

Execute refind-install with the option --preloader */path/to/preloader*

```
# refind-install --preloader /usr/share/preloader-signed/PreLoader.efi
```

Next time you boot with Secure Boot enabled, HashTool will launch and you will need to enroll the hash of rEFInd (loader.efi), rEFInd's drivers (e.g. ext4_x64.efi) and kernel (e.g. vmlinuz-linux).

See **refind-install(8) (https://man.archlinux.org/man/refind-install.8)** for more information.

> **Tip:** The signed *HashTool* is only capable of accessing the partition it was launched from. This means if your kernel is not on the ESP, you will not be able to enroll its hash from *HashTool*. You can workaround this by using **#KeyTool**, since it is capable of enrolling a hash in MokList and is not limited to one partition. Remember to enroll *KeyTool's* hash before using it.

Using shim

> **Warning:** Since version 15.3, EFI binaries without a valid .sbat section would not be launched by shim. This has not been supported by rEFInd yet. Please see the **SBAT documentation (https://github.com/rhboot/shim/blob/main/SBAT.md)** and the **discussion about rEFInd (https://sourceforge.net/p/refind/discussion/general/thread/c54261c145/)** for details.

Install **shim-signed (https://aur.archlinux.org/packages/shim-signed/)**AUR. Read **Secure Boot#shim**, but skip all file copying.

Using hashes

To use only hashes with *shim*, execute refind-install with the option --shim */path/to/shim*

```
# refind-install --shim /usr/share/shim-signed/shimx64.efi
```

Next time you boot with Secure Boot enabled, MokManager will launch and you will need to enroll the hash of rEFInd (grubx64.efi), rEFInd's drivers (e.g. ext4_x64.efi) and kernel (e.g. vmlinuz-linux).

Using Machine Owner Key

To sign rEFInd with a Machine Owner Key (MOK), install **sbsigntools (https://archlinux.org/packages/?name=sbsigntools)**.

> **Tip:** If you already have **created a MOK**, place the files in the directory /etc/refind.d/keys with the names refind_local.key (PEM format private key), refind_local.crt (PEM format certificate) and refind_local.cer (DER format certificate).

Execute refind-install with the options --shim */path/to/shim* and --localkeys :

```
# refind-install --shim /usr/share/shim-signed/shimx64.efi --localkeys
```

refind-install will create the keys for you and sign itself and its drivers. You will need to sign the kernel with the same key, e.g.:

```
# sbsign --key /etc/refind.d/keys/refind_local.key --cert /etc/refind.d/keys/refind_local.crt --output /boot/vmlinuz-linux /boot/vmlinuz-linux
```

> **Tip:** The kernel signing can be automated with a **pacman hook**, see **Secure Boot#Signing the kernel with a pacman hook**.

Once in MokManager add refind_local.cer to MoKList. refind_local.cer can be found inside a directory called keys in the rEFInd's installation directory, e.g. *esp*/EFI/refind/keys/refind_local.cer .

See **refind-install(8) (https://man.archlinux.org/man/refind-install.8)** for more information.

Using your own keys

Follow **Secure Boot#Using your own keys** to create keys.

Create directory /etc/refind.d/keys and place Signature Database (**db**) key and certificates in it. Name the files: refind_local.key (PEM format private key), refind_local.crt (PEM format certificate) and refind_local.cer (DER format certificate).

When running install script add option --localkeys , e.g.:

```
# refind-install --localkeys
```

rEFInd EFI binaries will be signed with the supplied key and certificate.

Manual installation

> **Tip:** rEFInd can boot Linux in many ways. See **The rEFInd Boot Manager: Methods of Booting Linux (https://www.rodsbooks.com/refind/linux.html)** for coverage of the various approaches.

If the refind-install script does not work for you, rEFInd can be set up manually.

First, copy the executable to the ESP:

```
# mkdir -p esp/EFI/refind
# cp /usr/share/refind/refind_x64.efi esp/EFI/refind/
```

If you want to install rEFInd to the default/fallback boot path replace *esp*/EFI/refind/ with *esp*/EFI/BOOT/ in the following instructions and copy rEFInd EFI executable to *esp*/EFI/BOOT/bootx64.efi :

```
# mkdir -p esp/EFI/BOOT
# cp /usr/share/refind/refind_x64.efi esp/EFI/BOOT/bootx64.efi
```

Then use **efibootmgr** to create a boot entry in the UEFI NVRAM, where *`/dev/sdX`* and *`Y`* are the device and partition number of your EFI system partition. If you are installing rEFInd to the default/fallback boot path *`esp`*`/EFI/BOOT/bootx64.efi`, you can skip this step.

```
# efibootmgr --create --disk /dev/sdX --part Y --loader /EFI/refind/refind_x64.efi --label "rEFInd Boot Manager" --verbose
```

At this point, you should be able to reboot into rEFInd, but it will not be able to boot your kernel. If your kernel does not reside on your ESP, rEFInd can mount your partitions to find it - provided it has the right drivers.

rEFInd automatically loads all drivers from the subdirectories `drivers` and `drivers_`*`arch`* (e.g. `drivers_x64`) in its install directory.

```
# mkdir esp/EFI/refind/drivers_x64
# cp /usr/share/refind/drivers_x64/drivername_x64.efi esp/EFI/refind/drivers_x64/
```

Now rEFInd should have a boot entry for your kernel, but it will not pass the correct kernel parameters. Set up **#Passing kernel parameters**. You should now be able to boot your kernel using rEFInd. If you are still unable to boot or if you want to tweak rEFInd's settings, many options can be changed with a configuration file:

```
# cp /usr/share/refind/refind.conf-sample esp/EFI/refind/refind.conf
```

The sample configuration file is well commented and self-explanatory.

Unless you have set `textonly` in the configuration file, you should copy rEFInd's icons to get rid of the ugly placeholders:

```
# cp -r /usr/share/refind/icons esp/EFI/refind/
```

You can try out different fonts by copying them and changing the `font` setting in `refind.conf`:

```
# cp -r /usr/share/refind/fonts esp/EFI/refind/
```

Tip: Pressing `F10` in rEFInd will save a screenshot to the top level directory of the ESP.

Upgrading

Pacman updates the rEFInd files in `/usr/share/refind/` and will not copy new files to the ESP for you. If `refind-install` worked for your original installation of rEFInd, you can rerun it to copy the updated files. The new configuration file will be copied as `refind.conf-sample` so that you can integrate changes into your existing configuration file using a **diff** tool. If your rEFInd required **#Manual installation**, you will need to figure out which files to copy yourself.

Pacman hook

You can automate the update process using a **pacman hook**:

```
/etc/pacman.d/hooks/refind.hook
--------------------------------
[Trigger]
Operation=Upgrade
Type=Package
Target=refind
```

```
[Action]
Description = Updating rEFInd on ESP
When=PostTransaction
Exec=/usr/bin/refind-install
```

Where the `Exec=` may need to be changed to the correct update command for your setup. If you did **#Manual installation**, you could create your own update script to call with the hook.

Tip: If you setup rEFInd with **#Secure Boot**, in addition to adding `--localkeys`, you may also want to add the option `--yes` to the `refind-install` command. It will prevent the command from failing if it gets executed when Secure Boot is disabled. See **refind-install(8) (https://man.archlinux.org/man/refind-install.8)** for more information.

Note: If the ESP is not mounted to `/boot` and you rely on automounting to mount it, make sure to preload the `vfat` module as instructed in **EFI system partition#Alternative mount points**. Otherwise, if **refind (https://archlinux.org/packages/?name=refind)** is upgraded together with the kernel, the ESP will become inaccessible.

Configuration

The rEFInd configuration `refind.conf` is located in the same directory as the rEFInd EFI application (usually *esp*/EFI/refind or *esp*/EFI/BOOT). The default configuration file contains extensive comments explaining all its options, see **Configuring the Boot Manager (https://www.rodsbooks.com/refind/configfile.html)** for more detailed explanations.

Passing kernel parameters

There are two methods for setting the **kernel parameters** that rEFInd will pass to the kernel.

For kernels automatically detected by rEFInd

For automatically detected kernels you can either specify the kernel parameters explicitly in `/boot/refind_linux.conf` or rely on rEFInd's ability to identify the root partition and kernel parameters. See **Methods of Booting Linux: For Those With Foresight or Luck: The Easiest Method (https://www.rodsbooks.com/refind/linux.html#easiest)** for more information.

Tip:

- rEFInd will automatically choose the Arch Linux icon (`os_arch.png`) for the boot entry when `/etc/os-release` is on the same partition as the kernel. If your `/boot` is a separate partition see **Configuring the Boot Manager: Setting OS Icons (https://www.rodsbooks.com/refind/configfile.html#icons)**.
- rEFInd **does not support detecting the distribution of unified kernel images (https://sourceforge.net/p/refind/discussion/general/thread/c3865a4e3a/)**. To have a icon for a **unified kernel image**, copy `/usr/share/refind/icons/os_arch.png` to *esp*/EFI/Linux/ and make sure the file names match. E.g., if you have *esp*/EFI/Linux/Arch-linux.efi, then name the icon— *esp*/EFI/Linux/Arch-linux.png.

For rEFInd to support the naming scheme of Arch Linux **kernels** and thus allow matching them with their respective initramfs images, you must uncomment and edit `extra_kernel_version_strings` option in `refind.conf`. E.g.:

esp/EFI/refind/refind.conf

```
...
extra_kernel_version_strings linux-hardened,linux-zen,linux-lts,linux
...
```

Note:

- rEFInd only supports detecting one initramfs image per kernel, meaning it will not detect fallback initramfs nor

microcode images. They must be specified manually.
- Without the above extra_kernel_version_strings line, the %v variable in refind_linux.conf will not work for Arch Linux kernels.

refind_linux.conf

If rEFInd automatically detects your kernel, you can place a refind_linux.conf file containing the kernel parameters in the same directory as your kernel. You can use /usr/share/refind/refind_linux.conf-sample as a starting point. The first uncommented line of refind_linux.conf will be the default parameters for the kernel. Subsequent lines will create entries in a submenu accessible using + , F2 , or Insert .

Alternatively, try running mkrlconf as root. It will attempt to find your kernel in /boot and automatically generate refind_linux.conf . The script will only set up the most basic kernel parameters, so be sure to check the file it created for correctness.

If you do not specify an initrd= parameter, rEFInd will automatically add it by searching for common RAM disk filenames in the same directory as the kernel. If you need multiple initrd= parameters, you must specify them manually in refind_linux.conf . For example, a microcode passed before the initramfs:

/boot/refind_linux.conf

```
"Boot using default options"     "root=PARTUUID=XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX rw add_efi_memmap initrd=boot\intel-ucode.img initrd=boot\amd-ucode.img initrd=boot\initramfs-%v.img"
"Boot using fallback initramfs"  "root=PARTUUID=XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX rw add_efi_memmap initrd=boot\intel-ucode.img initrd=boot\amd-ucode.img initrd=boot\initramfs-%v-fallback.img"
"Boot to terminal"               "root=PARTUUID=XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX rw add_efi_memmap initrd=boot\intel-ucode.img initrd=boot\amd-ucode.img initrd=boot\initramfs-%v.img systemd.unit=multi-user.target"
```

Warning:

- initrd path is relative to the root of the file system on which the kernel resides. This could be initrd=\boot\initramfs-%v.img or, if /boot is a separate partition (e.g. the ESP), initrd=initramfs-%v.img .
- Use backslashes (\ ) as path separators in the initrd parameter, otherwise the kernel may fail to find the initramfs image(s): EFI stub: ERROR: Failed to open file: /boot/intel-ucode.img .
- If using Booster generated initramfs images, replace initramfs with booster in the initramfs files name. E.g. initrd=\boot\booster-%v.img .

Note: rEFInd replaces %v in refind_linux.conf with the kernel's version (by extracting if from the file name). For rEFInd to support Arch Linux kernels, the extra_kernel_version_strings in esp/EFI/refind/refind.conf must be edited as instructed in #For kernels automatically detected by rEFInd.

Without configuration

If you merely install rEFInd onto the ESP and launch it without any further ado (say via UEFI shell or KeyTool, or directly from firmware) you still get a menu to boot from via autodetection, with no configuration required whatsoever.

This works because rEFInd has a fallback mechanism that can:

- Identify the root partition (for root= parameter) via the Discoverable Partitions Specification (https://systemd.io/DISCOVERABLE_PARTITIONS/) or /etc/fstab .
- Detect kernel options (ro or rw) from GPT partition attributes (using attribute 60 "read-only") or /etc/fstab .

Note: rEFInd does not support escape codes (e.g. for spaces) in /etc/fstab .

For manual boot stanzas

If your kernel is not autodetected, or if you simply want more control over the options for a menu entry, you can manually create boot entries using stanzas in `refind.conf`. Ensure that `scanfor` includes `manual` or these entries will not appear in rEFInd's menu. Kernel parameters are set with the `options` keyword. rEFInd will append the `initrd=` parameter using the file specified by the `initrd` keyword in the stanza. If you need additional initrds (e.g. for **Microcode**), you can specify them in `options` (and the one specified by the `initrd` keyword will be added to the end).

Manual boot stanzas are explained in **Creating Manual Boot Stanzas (https://www.rodsbooks.com/refind/configfile.html#stanzas)**.

esp/EFI/refind/refind.conf

```
...

menuentry "Arch Linux" {
    icon    /EFI/refind/icons/os_arch.png
    volume  "Arch Linux"
    loader  /boot/vmlinuz-linux
    initrd  /boot/initramfs-linux.img
    options "root=PARTUUID=XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX rw add_efi_memmap initrd=boot\intel-ucode.img initrd=boot\amd-ucode.img"
    submenuentry "Boot using fallback initramfs" {
        initrd /boot/initramfs-linux-fallback.img
    }
    submenuentry "Boot to terminal" {
        add_options "systemd.unit=multi-user.target"
    }
}
```

It is likely that you will need to change `volume` to match either a filesystem's LABEL, a PARTLABEL, or a PARTUUID of the partition where the kernel image resides. The PARTUUID needs to be in uppercase. See **Persistent block device naming#by-label** for examples of assigning a volume label. If `volume` is not specified it defaults to volume from which rEFInd was launched (typically EFI system partition).

Warning:

- `loader` and `initrd` paths are relative to the root of `volume`. If `/boot` is a separate partition (e.g. the ESP), the loader and initrd paths would be `/vmlinuz-linux` and `/initramfs-linux.img`, respectively.
- Use backslashes (`\`) as path separators in all quoted `initrd` parameters, otherwise the kernel may fail to find the initramfs image(s): `EFI stub: ERROR: Failed to open file: /boot/initramfs-linux.img`.
- If using **Booster** generated initramfs images, replace `initramfs` with `booster` in the initramfs files name. E.g. `initrd /boot/booster-linux.img`.

Installation alongside an existing UEFI Windows installation

Note: The usual caveats of **Dual boot with Windows** apply.

rEFInd is compatible with the EFI system partition created by a UEFI Windows installation, so there is no need to create or format another FAT32 partition when installing Arch alongside Windows. Simply mount the existing ESP and install rEFInd as usual. By default, rEFInd's autodetection feature should recognize any existing Windows/recovery bootloaders.

Note: In some cases, Windows behaves differently (low resolution boot screen, OEM logo replaced by Windows logo, black screen after boot screen, artifacting). If you face such issues, try setting `use_graphics_for +,windows` in *esp*/EFI/refind/refind.conf or adding `graphics on` to the Windows boot stanza.

Tools

rEFInd supports running various **3rd-party tools (https://www.rodsbooks.com/refind/installing.html#addons)**. Tools need to be installed separately. Edit `showtools` in `refind.conf` to choose which ones to show.

esp/EFI/refind/refind.conf

```
...
showtools shell, memtest, mok_tool, gdisk, fwupdate ...
...
```

UEFI shell

See **Unified Extensible Firmware Interface#UEFI Shell**.

Copy `shellx64.efi` to the root of the **EFI system partition**.

Memtest86

Install **memtest86-efi (https://aur.archlinux.org/packages/memtest86-efi/)**AUR and copy it to *esp*/EFI/tools/ .

```
# cp /usr/share/memtest86-efi/bootx64.efi esp/EFI/tools/memtest86.efi
```

Key management tools

rEFInd can detect Secure Boot key management tools if they are placed in rEFInd's directory on ESP, *esp*/ or *esp*/EFI/tools/ .

HashTool

Follow **#Using PreLoader** and `HashTool.efi` will be placed in rEFInd's directory.

MokManager

Follow **#Using shim** and MokManager will be placed in rEFInd's directory.

KeyTool

Install **efitools (https://archlinux.org/packages/?name=efitools)**.

Place KeyTool EFI binary in *esp*/ or *esp*/EFI/tools/ with the name `KeyTool.efi` or `KeyTool-signed.efi` .

See **Secure Boot#Using KeyTool** for instructions on signing `KeyTool.efi` .

GPT fdisk (gdisk)

Download the **gdisk EFI application** and copy `gdisk_x64.efi` to *esp*/EFI/tools/ .

fwupdate

Install and setup **fwupd**.

Copy the `fwupx64.efi` binary and firmware file to *esp*/EFI/tools/ :

```
# cp /usr/lib/fwupd/efi/fwupdx64.efi esp/EFI/tools/
```

Poweroff or reboot

rEFInd reportedly have poweroff and reboot menu entries built in. Since this list of tools is the most extensive of its kind in this wiki, users of UEFI shell, or other UEFI boot managers, such as systemd-boot, might be interested in poweroffforreboot.efi (https://aur.archlinux.org/packages/poweroffforreboot.efi/)AUR.

Tips and tricks

Using drivers in UEFI shell

To use rEFInd's drivers in UEFI shell load them using command `load` and refresh mapped drives with `map -r`.

```
Shell> load FS0:\EFI\refind\drivers\ext4_x64.efi
Shell> map -r
```

Now you can access your file system from UEFI shell.

Setting efifb resolution

If the resolution in `refind.conf` is set to an incorrect value, on all systems except Apple Macs rEFInd will display a list of supported resolutions. For Apple Macs it will silently use the default resolution.

To determine framebuffer resolutions supported by **efifb (https://www.kernel.org/doc/html/latest/fb/efifb.html)**, copy `/usr/share/gnu-efi/apps/x86_64/modelist.efi` from gnu-efi (https://archlinux.org/packages/?name=gnu-efi) to the root of **ESP**. Enter the **UEFI shell** and run `modelist.efi`.

```
Shell> FS0:\modelist.efi
```

```
GOP reports MaxMode 3
 0: 640x480 BGRR pitch 640
*1: 800x600 BGRR pitch 800
 2: 1024x768 BGRR pitch 1024
```

Set one in `refind.conf`. Reboot and check if settings has been applied by running `dmesg | grep efifb` as root.

Btrfs subvolume support

Tip: make sure `btrfs_x64.efi` driver is installed, it can be installed manually by copying from `/usr/share/refind/drivers_x64/btrfs_x64.efi` to *esp*`/EFI/refind/drivers_x64/btrfs_x64.efi`, or you can install all drivers with the `refind-install /dev/sdx --alldrivers` option.

Warning: `btrfs_x64.efi` does not support `raid1c3/4`.

Auto detection

To allow kernel auto detection on a Btrfs subvolume uncomment and edit `also_scan_dirs` in `refind.conf`.

```
esp/EFI/refind/refind.conf
```

```
...
also_scan_dirs +,subvolume/boot
...
```

Next add `subvol=`*`subvolume`* to `rootflags` in `refind_linux.conf` and then prepend *`subvolume`* to the initrd path.

/boot/refind_linux.conf

```
"Boot using standard options"  "root=PARTUUID=XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX rw rootflags=subvol=subvolume initrd=subvolume\boot\initramfs-%v.img"
```

Manual boot stanza

If booting a **btrfs** subvolume as root, prepend the path to the subvolume to the loader and initrd paths, and amend the `options` line with `rootflags=subvol=root_subvolume`. In the example below, root has been mounted as a btrfs subvolume called 'ROOT' (e.g. `mount -o subvol=ROOT /dev/sdxY /mnt`):

esp/EFI/refind/refind.conf

```
...
menuentry "Arch Linux" {
      icon    /EFI/refind/icons/os_arch.png
      volume   "[bootdevice]"
      loader   /ROOT/boot/vmlinuz-linux
      initrd   /ROOT/boot/initramfs-linux.img
      options  "root=PARTUUID=XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX rw rootflags=subvol=ROOT"
...
}
```

A failure to do so will otherwise result in the following error message: `ERROR: Root device mounted successfully, but /sbin/init does not exist.`

LoaderDevicePartUUID

Since version 0.13.1, rEFInd supports setting the UEFI variable **LoaderDevicePartUUID (https://systemd.io/BOOT_LOADER_INTERFACE/)**. Enabling this allows **systemd-gpt-auto-generator(8) (https://man.archlinux.org/man/systemd-gpt-auto-generator.8)** to automount the EFI system partition without needing to specify it in `/etc/fstab`. See **systemd#GPT partition automounting**.

For rEFInd to set `LoaderDevicePartUUID`, edit `refind.conf` and uncomment `write_systemd_vars true`:

esp/EFI/refind/refind.conf

```
...
write_systemd_vars true
...
```

You can verify if it is set by checking its value with `cat /sys/firmware/efi/efivars/LoaderDevicePartUUID-4a67b082-0a4c-41cf-b6c7-440b29bb8c4f` or by looking at the state of "Boot loader sets ESP information" in `bootctl` output.

Troubleshooting

Apple Macs

mactel-boot (https://aur.archlinux.org/packages/mactel-boot/)[AUR] is an experimental *bless* utility for Linux. If that does not work, use *bless* from within OSX to set rEFInd as the default boot entry:

```
# bless --setBoot --folder esp/EFI/refind --file esp/EFI/refind/refind_x64.efi
```

VirtualBox

VirtualBox before version 6.1 will only boot the default *esp*/EFI/BOOT/bootx64.efi path, so refind-install needs to be used with at least the --usedefault option. See **VirtualBox/Install Arch Linux as a guest#Installation in EFI mode on VirtualBox < 6.1** for more information.

Blank rEFInd menu screen

If your drivers_x64 folder contains multiple file system drivers (see **#Installing the rEFInd Boot Manager** for clarification), this can lead to an improper functioning of rEFInd through a file system driver bug, whereby only a blank screen and with the rEFInd logo is shown (for custom themes, this would be the set background image). To fix this, simply remove all drivers **except the one for the file system on which the kernel resides**.

Another potential blank screen cause occurs when dual booting with Windows, where rEFInd is unsuccessful in auto-scanning the EFI system partitions on other disks. To remedy this, use *blkid* to identify Windows partitions, and add the **PARTUUID** of each Windows partition as a comma-separated entry to the variable dont_scan_volumes in refind.conf . For example:

```
# blkid
```

```
/dev/nvme0n1p1: LABEL="SYSTEM" UUID="4CE7-C215" BLOCK_SIZE="512" TYPE="vfat" PARTLABEL="EFI system partition" PARTUUID="13aa9955-1234-5678-9098-006c334b5088"
/dev/nvme0n1p5: LABEL="Windows RE Tools" BLOCK_SIZE="512" UUID="08C4E6C5C4E6B45A" TYPE="ntfs" PARTLABEL="Basic data partition" PARTUUID="4eced110-0987-6543-2123-b0ab8576869b"
```

esp/EFI/refind/refind.conf

```
...
dont_scan_volumes 13aa9955-1234-5678-9098-006c334b5088,4eced110-0987-6543-2123-b0ab8576869b
...
```

See also

- **The rEFInd Boot Manager (https://www.rodsbooks.com/refind/)** by Roderick W. Smith.
- **Wikipedia:rEFInd**
- /usr/share/refind/docs/README.txt
- **rEFInd discussion forum on Sourceforge (https://sourceforge.net/p/refind/discussion/)**

Retrieved from "**https://wiki.archlinux.org/index.php?title=REFInd&oldid=707429**"

This page was last edited on 24 December 2021, at 13:59.

Content is available under GNU Free Documentation License 1.3 or later unless otherwise noted.

- Privacy policy
- About ArchWiki
- Disclaimers