OpenCore

Reference Manual (~~0.7.9~~0.8.0)

[2022.03.20]

Copyright ©2018-2022 vit9696

Requirement: 10.4
Description: Disables primary checksum (`0x58-0x59`) writing in AppleRTC.

Note 1: This option will not protect other areas from being overwritten, see RTCMemoryFixup kernel extension if this is desired.

Note 2: This option will not protect areas from being overwritten at firmware stage (e.g. macOS bootloader), see `AppleRtcRam` protocol description if this is desired.

9. `ExtendBTFeatureFlags`
Type: `plist boolean`
Failsafe: `false`
Requirement: 10.8-11
Description: Set `FeatureFlags` to `0x0F` for full functionality of Bluetooth, including Continuity.

Note: This option is a substitution for BT4LEContinuityFixup.kext, which does not function properly due to late patching progress.

10. `ExternalDiskIcons`
Type: `plist boolean`
Failsafe: `false`
Requirement: 10.4
Description: Apply icon type patches to AppleAHCIPort.kext to force internal disk icons for all AHCI disks.

Note: This option should be avoided whenever possible. Modern firmware typically have compatible AHCI controllers.

11. `ForceAquantiaEthernet`
Type: `plist boolean`
Failsafe: `false`
Requirement: 10.15.4
Description: Enable Aquantia AQtion AQC-107s based 10GbE network cards support.

This option enables Aquantia AQtion AQC-107s based 10GbE network cards support, which used to work natively before macOS 10.15.4.

12. `ForceSecureBootScheme`
Type: `plist boolean`
Failsafe: `false`
Requirement: 11
Description: Force `x86` scheme for IMG4 verification.

Note: This option is required on virtual machines when using `SecureBootModel` different from `x86legacy`.

13. `IncreasePciBarSize`
Type: `plist boolean`
Failsafe: `false`
Requirement: 10.10
Description: Allows IOPCIFamily to boot with 2 GB PCI BARs.

Normally macOS restricts PCI BARs to 1 GB. Enabling this option (still) does not let macOS actually use PCI devices with larger BARs.

Note: This option should be avoided whenever possible. A need for this option indicates misconfigured or defective firmware.

14. `LapicKernelPanic`
Type: `plist boolean`
Failsafe: `false`
Requirement: 10.6 (64-bit)
Description: Disables kernel panic on LAPIC interrupts.

15. `LegacyCommpage`
Type: `plist boolean`
Failsafe: `false`

Requirement: 10.4 - 10.6
Description: Replaces the default 64-bit commpage bcopy implementation with one that does not require SSSE3, useful for legacy platforms. This prevents a `commpage no match for last` panic due to no available 64-bit bcopy functions that do not require SSSE3.

16. `PanicNoKextDump`
    Type: `plist boolean`
    Failsafe: `false`
    Requirement: 10.13 (not required for older)
    Description: Prevent kernel from printing kext dump in the panic log preventing from observing panic details. Affects 10.13 and above.

17. `PowerTimeoutKernelPanic`
    Type: `plist boolean`
    Failsafe: `false`
    Requirement: 10.15 (not required for older)
    Description: Disables kernel panic on setPowerState timeout.

    An additional security measure was added to macOS Catalina (10.15) causing kernel panic on power change timeout for Apple drivers. Sometimes it may cause issues on misconfigured hardware, notably digital audio, which sometimes fails to wake up. For debug kernels `setpowerstate_panic=0` boot argument should be used, which is otherwise equivalent to this quirk.

18. `ProvideCurrentCpuInfo`
    Type: `plist boolean`
    Failsafe: `false`
    Requirement: 10.8 (10.14)
    Description: Provides current CPU info to the kernel.

    This quirk works differently depending on the CPU:

    - For Microsoft Hyper-V it provides the correct TSC and FSB values to the kernel, as well as disables CPU topology validation (10.8+).
    - For KVM and other hypervisors it provides precomputed MSR 35h values solving kernel panic with `-cpu host`.
    - For Intel CPUs it adds support for asymmetrical SMP systems (e.g. Intel Alder Lake) by patching core count to thread count along with the supplemental required changes (10.14+).

19. `SetApfsTrimTimeout`
    Type: `plist integer`
    Failsafe: `-1`
    Requirement: 10.14 (not required for older)
    Description: Set trim timeout in microseconds for APFS filesystems on SSDs.

    The APFS filesystem is designed in a way that the space controlled via the spaceman structure is either used or free. This may be different in other filesystems where the areas can be marked as used, free, and *unmapped*. All free space is trimmed (unmapped/deallocated) at macOS startup. The trimming procedure for NVMe drives happens in LBA ranges due to the nature of the `DSM` command with up to 256 ranges per command. The more fragmented the memory on the drive is, the more commands are necessary to trim all the free space.

    Depending on the SSD controller and the level of drive fragmenation, the trim procedure may take a considerable amount of time, causing noticeable boot slowdown. The APFS driver explicitly ignores previously unmapped areas and repeatedly trims them on boot. To mitigate against such boot slowdowns, the macOS driver introduced a timeout (`9.999999` seconds) that stops the trim operation when not finished in time.

    On several controllers, such as Samsung, where the deallocation process is relatively slow, this timeout can be reached very quickly. Essentially, it means that the level of fragmentation is high, thus macOS will attempt to trim the same lower blocks that have previously been deallocated, but never have enough time to deallocate higher blocks. The outcome is that trimming on such SSDs will be non-functional soon after installation, resulting in additional wear on the flash.

    One way to workaround the problem is to increase the timeout to an extremely high value, which at the cost of slow boot times (extra minutes) will ensure that all the blocks are trimmed. ~~Set~~ Setting this option to a

high value, such as `4294967295` ~~, to ensure~~ (a.k.a. `-1`) ensures that all blocks are trimmed. Alternatively, use over-provisioning, if supported, or create a dedicated unmapped partition where the reserve blocks can be found by the controller. Conversely, the trim operation can be mostly disabled by setting a very low timeout value~~. e.g. 999.~~, while `0` entirely disables it. Refer to this article for details.

On macOS 12+, it is no longer possible to ~~set~~ specify trim timeout for APFS filesystems. However, ~~trim~~ it can be disabled ~~when the timeout value is~~ by setting `0`.

Note: System Information may still report `TRIM Support: Yes` with this option set to `0`. Yet, this is false positive and trim is de facto disabled.

20. `ThirdPartyDrives`
Type: `plist boolean`
Failsafe: `false`
Requirement: 10.6 (not required for older)
Description: Apply vendor patches to IOAHCIBlockStorage.kext to enable native features for third-party drives, such as TRIM on SSDs or hibernation support on 10.15 and newer.

Note: This option may be avoided on user preference. NVMe SSDs are compatible without the change. For AHCI SSDs on modern macOS version there is a dedicated built-in utility called `trimforce`. Starting from 10.15 this utility creates `EnableTRIM` variable in `APPLE_BOOT_VARIABLE_GUID` namespace with `01 00 00 00` value.

21. `XhciPortLimit`
Type: `plist boolean`
Failsafe: `false`
Requirement: 10.11 (not required for older)
Description: Patch various kexts (AppleUSBXHCI.kext, AppleUSBXHCIPCI.kext, IOUSBHostFamily.kext) to remove USB port count limit of 15 ports.

Note: This option should be avoided whenever possible. USB port limit is imposed by the amount of used bits in locationID format and there is no possible way to workaround this without heavy OS modification. The only valid solution is to limit the amount of used ports to 15 (discarding some). More details can be found on AppleLife.ru.

7.9 Scheme Properties

These properties are particularly relevant for older macOS operating systems. Refer to the Legacy Apple OS section for details on how to install and troubleshoot such macOS installations.

1. `CustomKernel`
Type: `plist boolean`
Failsafe: `false`
Description: Use customised kernel cache from the `Kernels` directory located at the root of the ESP partition.

Unsupported platforms including `Atom` and `AMD` require modified versions of XNU kernel in order to boot. This option provides the possibility to using a customised kernel cache which contains such modifications from ESP partition.

2. `FuzzyMatch`
Type: `plist boolean`
Failsafe: `false`
Description: Use `kernelcache` with different checksums when available.

On macOS 10.6 and earlier, `kernelcache` filename has a checksum, which essentially is `adler32` from SMBIOS product name and EfiBoot device path. On certain firmware, the EfiBoot device path differs between UEFI and macOS due to ACPI or hardware specifics, rendering `kernelcache` checksum as always different.

This setting allows matching the latest `kernelcache` with a suitable architecture when the `kernelcache` without suffix is unavailable, improving macOS 10.6 boot performance on several platforms.

3. `KernelArch`
Type: `plist string`
Failsafe: `Auto` (Choose the preferred architecture automatically)
Description: Prefer specified kernel architecture (`i386`, `i386-user32`, `x86_64`) when available.

Development and debug kernels produce more useful kernel panic logs. Consider downloading and installing the `KernelDebugKit` from developer.apple.com when debugging a problem. To activate a development kernel, the boot argument `kcsuffix=development` should be added. Use the `uname -a` command to ensure that the current loaded kernel is a development (or a debug) kernel.

In cases where the OpenCore kernel panic saving mechanism is not used, kernel panic logs may still be found in the `/Library/Logs/DiagnosticReports` directory.

Starting with macOS Catalina, kernel panics are stored in JSON format and thus need to be preprocessed before passing to `kpdescribe.sh`:

```
cat Kernel.panic | grep macOSProcessedStackshotData |
- python -c 'import json,sys;print(json.load(sys.stdin)["macOSPanicString"])'
+ python3 -c 'import json,sys;print(json.load(sys.stdin)["macOSPanicString"])'
```

3. `DisableWatchDog`
   Type: `plist boolean`
   Failsafe: `false`
   Description: Some types of firmware may not succeed in booting the operating system quickly, especially in debug mode. This results in the watchdog timer aborting the process. This option turns off the watchdog timer.

4. `DisplayDelay`
   Type: `plist integer`
   Failsafe: `0`
   Description: Delay in microseconds executed after every printed line visible onscreen (i.e. console).

5. `DisplayLevel`
   Type: `plist integer`, 64 bit
   Failsafe: `0`
   Description: EDK II debug level bitmask (sum) showed onscreen. Unless `Target` enables console (onscreen) printing, onscreen debug output will not be visible.

   The following levels are supported (discover more in DebugLib.h):

   - `0x00000002` (bit `1`) — `DEBUG_WARN` in `DEBUG`, `NOOPT`, `RELEASE`.
   - `0x00000040` (bit `6`) — `DEBUG_INFO` in `DEBUG`, `NOOPT`.
   - `0x00400000` (bit `22`) — `DEBUG_VERBOSE` in custom builds.
   - `0x80000000` (bit `31`) — `DEBUG_ERROR` in `DEBUG`, `NOOPT`, `RELEASE`.

6. `LogModules`
   Type: `plist string`
   Failsafe: `*`
   Description: Filter log entries by module.

   This option filters logging generated by specific modules, both in the log and onscreen. Two modes are supported:

   - `+` — Positive filtering: Only present selected modules.
   - `-` — Negative filtering: Exclude selected modules.

   When multiple ones are selected, comma (`,`) should be used as the splitter. For instance, `+OCCPU,OCA,OCB` means *only* `OCCPU`, `OCA`, `OCB` being printed, while `-OCCPU,OCA,OCB` indicates these modules being filtered out (i.e. *not* logged). When no symbol is specified, positive filtering (`+`) will be used. `*` indicates all modules being logged.

   Note 1: Acronyms of libraries can be found in the `Libraries` section below.

   Note 2: Messages printed before the configuration of log protocol cannot be filtered.

7. `SerialInit`
   Type: `plist boolean`
   Failsafe: `false`
   Description: Perform serial port initialisation.

   This option will perform serial port initialisation within OpenCore prior to enabling (any) debug logging. Serial port configuration is defined via PCDs at compile time in `gEfiMdeModulePkgTokenSpaceGuid` GUID.

Older boards like ICH6 may not always have HPET setting in the firmware preferences, this option tries to force enable it.

2. `EnableVectorAcceleration`
   Type: `plist boolean`
   Failsafe: `false`
   Description: Enable AVX vector acceleration of SHA-512 and SHA-384 hashing algorithms.

   Note: This option may cause issues on certain laptop firmwares, including Lenovo.

3. `EnableVmx`
   Type: `plist boolean`
   Failsafe: `false`
   Description: Enable Intel virtual machine extensions.

   Note: Required to allow virtualization in Windows on some Mac hardware. VMX is enabled or disabled and locked by BIOS before OpenCore starts on most firmware. Use BIOS to enable virtualization where possible.

4. `DisableSecurityPolicy`
   Type: `plist boolean`
   Failsafe: `false`
   Description: Disable platform security policy.

   Note: This setting disables various security features of the firmware, defeating the purpose of any kind of Secure Boot. Do NOT enable if using UEFI Secure Boot.

5. `ExitBootServicesDelay`
   Type: `plist integer`
   Failsafe: `0`
   Description: Adds delay in microseconds after `EXIT_BOOT_SERVICES` event.

   This is a very rough workaround to circumvent the `Still waiting for root device` message on some APTIO IV firmware (ASUS Z87-Pro) particularly when using FileVault 2. It appears that for some reason, they execute code in parallel to `EXIT_BOOT_SERVICES`, which results in the SATA controller being inaccessible from macOS. A better approach is required and Acidanthera is open to suggestions. Expect 3 to 5 seconds to be adequate when this quirk is needed.

6. `ForceOcWriteFlash`
   Type: `plist boolean`
   Failsafe: `false`
   Description: Enables writing to flash memory for all OpenCore-managed NVRAM system variables.

   Note: This value should be disabled on most types of firmware but is left configurable to account for firmware that may have issues with volatile variable storage overflows or similar. Boot issues across multiple OSes can be observed on e.g. Lenovo Thinkpad T430 and T530 without this quirk. Apple variables related to Secure Boot and hibernation are exempt from this for security reasons. Furthermore, some OpenCore variables are exempt for different reasons, such as the boot log due to an available user option, and the TSC frequency due to timing issues. When toggling this option, a NVRAM reset may be required to ensure full functionality.

7. `ForgeUefiSupport`
   Type: `plist boolean`
   Failsafe: `false`
   Description: Implement partial UEFI 2.x support on EFI 1.x firmware.

   This setting allows running some software written for UEFI 2.x firmware like NVIDIA GOP Option ROMs on hardware with older EFI 1.x firmware like `MacPro5,1`.

8. `IgnoreInvalidFlexRatio`
   Type: `plist boolean`
   Failsafe: `false`
   Description: Some types of firmware (such as APTIO IV) may contain invalid values in the `MSR_FLEX_RATIO` (`0x194`) MSR register. These values may cause macOS boot failures on Intel platforms.