NAME

gamemoded − daemon that optimises system performance on demand

SYNOPSIS

gamemoded [OPTIONS...]

DESCRIPTION

GameMode is a daemon/lib combo for Linux that allows games to request a set of optimisations be temporarily applied to the host OS.

The design has a clear cut abstraction between the host daemon and library (**gamemoded** and **libgamemode**), and the client loaders (**libgamemodeauto** and **gamemode_client.h**) that allows for safe usage without worrying about whether the daemon is installed or running. This design also means that while the host library currently relies on systemd for exchanging messages with the daemon, it's entirely possible to implement other internals that still work with the same clients.

GameMode was designed primarily as a stop-gap solution to problems with the Intel and AMD CPU powersave or ondemand governors, but is intended to be expanded beyond just CPU governor states, as there are a wealth of automation tasks one might want to apply.

OPTIONS

−r[PID], −−request=[PID]
Toggle gamemode for process. When no PID given, requests gamemode and pauses

−s[PID], −−status=[PID]
Query the status of gamemode for process When no PID given, queries the status globally

−d, −−daemonize
Run the daemon as a separate process (daemonize it)

−l, −−log-to-syslog
Log to syslog

−h, −−help
Print help text

−t, −−test
Run diagnostic tests on the current installation

−v, −−version
Print the version

USAGE

libgamemodeauto.so.0 can be pre-loaded into any program to request **gamemoded** begin or end the mode. See gamemoderun(1) for details.

The **gamemode_client.h** header can be used by developers to build the requests into a program:

```
#include "gamemode_client.h"

if( gamemode_request_start() < 0 ) {
   fprintf( stderr, "gamemode request failed: %s\n", gamemode_error_string() )
}

/* run the process */

/* Not required, gamemoded can clean up after game exits */
gamemode_request_end();
```

Atlernatively developers can define **GAMEMODE_AUTO** to mimic the behaviour of **libgamemodeauto.so.0**:

```
#define GAMEMODE_AUTO
#include "gamemode_client.h"
```

Or, distribute **libgamemodeauto.so.0** and either link with **−lgamemodeauto** or inject it as above with **LD_PRELOAD**.

CONFIG

gamemoded will load and merge **gamemode.ini** config files from these directories in the following order:

```
/usr/share/gamemode/
/etc/
$XDG_CONFIG_HOME or $HOME/.config/
$PWD
```

Behaviour of the config file can be explained by presenting a commented example:

```
[general]
; The reaper thread will check every 5 seconds for exited clients, for config file changes, and for the CPU/iGPU powe
reaper_freq=5

; The desired governor is used when entering GameMode instead of "performance"
desiredgov=performance
; The default governor is used when leaving GameMode instead of restoring the original value
;defaultgov=powersave

; The iGPU desired governor is used when the integrated GPU is under heavy load
igpu_desiredgov=powersave
; Threshold to use to decide when the integrated GPU is under heavy load.
; This is a ratio of iGPU Watts / CPU Watts which is used to determine when the
; integraged GPU is under heavy enough load to justify switching to
; igpu_desiredgov.  Set this to -1 to disable all iGPU checking and always
; use desiredgov for games.
igpu_power_threshold=0.3

; GameMode can change the scheduler policy to SCHED_ISO on kernels which support it (currently
; not supported by upstream kernels). Can be set to "auto", "on" or "off". "auto" will enable
; with 4 or more CPU cores. "on" will always enable. Defaults to "off".
softrealtime=off

; GameMode can renice game processes. You can put any value between 0 and 20 here, the value
; will be negated and applied as a nice value (0 means no change). Defaults to 0.
renice=0

; By default, GameMode adjusts the iopriority of clients to BE/0, you can put any value
; between 0 and 7 here (with 0 being highest priority), or one of the special values
; "off" (to disable) or "reset" (to restore Linux default behavior based on CPU priority),
; currently, only the best-effort class is supported thus you cannot set it here
ioprio=0

; Sets whether gamemode will inhibit the screensaver when active
; Defaults to 1
```

```
inhibit_screensaver=1

[filter]
; If "whitelist" entry has a value(s)
; gamemode will reject anything not in the whitelist
;whitelist=RiseOfTheTombRaider

; Gamemode will always reject anything in the blacklist
;blacklist=HalfLife3
;    glxgears

[gpu]
; Here Be Dragons!
; Warning: Use these settings at your own risk
; Any damage to hardware incurred due to this feature is your responsibility and yours alone
; It is also highly recommended you try these settings out first manually to find the sweet spots

; Setting this to the keyphrase "accept-responsibility" will allow gamemode to apply GPU optimisations such as over
;apply_gpu_optimisations=0

; The DRM device number on the system (usually 0), ie. the number in /sys/class/drm/card0/
;gpu_device=0

; Nvidia specific settings
; Requires the coolbits extension activated in nvidia-xconfig
; This corresponds to the desired GPUPowerMizerMode
; "Adaptive"=0 "Prefer Maximum Performance"=1 and "Auto"=2
; See NV_CTRL_GPU_POWER_MIZER_MODE and friends in https://github.com/NVIDIA/nvidia-settings/blob/ma
;nv_powermizer_mode=1

; These will modify the core and mem clocks of the highest perf state in the Nvidia PowerMizer
; They are measured as Mhz offsets from the baseline, 0 will reset values to default, -1 or unset will not modify value
;nv_core_clock_mhz_offset=0
;nv_mem_clock_mhz_offset=0

; AMD specific settings
; Requires a relatively up to date AMDGPU kernel module
; See: https://dri.freedesktop.org/docs/drm/gpu/amdgpu.html#gpu-power-thermal-controls-and-monitoring
; It is also highly recommended you use lm-sensors (or other available tools) to verify card temperatures
; This corresponds to power_dpm_force_performance_level, "manual" is not supported for now
;amd_performance_level=high

[supervisor]
; This section controls the new gamemode functions gamemode_request_start_for and gamemode_request_end_for
; The whilelist and blacklist control which supervisor programs are allowed to make the above requests
;supervisor_whitelist=
;supervisor_blacklist=

; In case you want to allow a supervisor to take full control of gamemode, this option can be set
; This will only allow gamemode clients to be registered by using the above functions by a supervisor client
;require_supervisor=0

[custom]
; Custom scripts (executed using the shell) when gamemode starts and ends
```

```
;start=notify-send "GameMode started"
;    /home/me/bin/stop_ethmining.sh

;end=notify-send "GameMode ended"
;    /home/me/bin/start_ethmining.sh

; Timeout for scripts (seconds). Scripts will be killed if they do not complete within this time.
;script_timeout=10
```

SEE ALSO

gamemoderun(1), systemd(1)

ABOUT

GameMode source can be found at *https://github.com/FeralInteractive/gamemode.git*

AUTHOR

Feral Interactive (linux-contact@feralinteractive.com)