Magic SysRq key

The **magic SysRq key** is a key combination understood by the Linux kernel, which allows the user to perform various low-level commands regardless of the system's state. It is often used to recover from freezes, or to reboot a computer without corrupting the filesystem.[1] Its effect is similar to the computer's hardware reset button (or power switch) but with many more options and much more control.

The SysRq key

This key combination provides access to powerful features for software development and disaster recovery. In this sense, it can be considered a form of escape sequence. Principal among the offered commands are means to forcibly unmount file systems, kill processes, recover keyboard state, and write unwritten data to disk. With respect to these tasks, this feature serves as a tool of last resort.

The magic SysRq key cannot work under certain conditions, such as a kernel panic[2] or a hardware failure preventing the kernel from running properly.

Contents

Commands

The key combination consists of Alt + Sys Req and another key, which controls the command issued. In Ubuntu the combination of Alt+SysRequest acts as "print screen" command.

The combinations always assume the QWERTY keyboard layout; for example, on a Dvorak Simplified Keyboard, the combination to shut the system down uses the R key instead of O. Furthermore, some keyboards may not provide a separate SysRq key. In this case, a separate PrtScr key should be present.

On some devices, notably laptops, the Fn key may need to be pressed to use the magic SysRq key.

The magic SysRq key may also be accessible from the serial console (by sending an RS-232 break signal) if the sysctl option is set to allow it.[3]

When logged in using SSH the SysRq may be accessible by writing to /proc/sysrq-trigger (echo s > /proc/sysrq-trigger).[1]

Action	QWERTY	Dvorak	AZERTY	Colemak
Set the console log level, which controls the types of kernel messages that are output to the console	**0** through **9**	**0** through **9**	**0** through **9** (*without* using shift)	**0** through **9**
Immediately reboot the system, without unmounting or syncing filesystems	**b**	**x**	**b**	**b**
Perform a system crash. A crashdump will be taken if it is configured.	**c**	**j**	**c**	**c**
Display all currently held Locks (CONFIG_LOCKDEP kernel option is required)	**d**	**e**	**d**	**s**
Send the SIGTERM signal to all processes except init (PID 1)	**e**	**.**	**e**	**f**
Call oom_kill, which kills a process to alleviate an OOM condition	**f**	**u**	**f**	**t**
When using Kernel Mode Setting, provides emergency support for switching back to the kernel's framebuffer console[4] If the in-kernel debugger 'kdb' is present, enter the debugger.	**g**	**i**	**g**	**d**
Output a terse help document to the console Any key which is not bound to a command should also perform this action	**h**	**d**	**h**	**h**
Send the SIGKILL signal to all processes except init	**i**	**c**	**i**	**u**
Forcibly "Just thaw it" – filesystems frozen by the FIFREEZE ioctl.	**j**	**h**	**j**	**n**
Kill all processes on the current virtual console (can kill X and svgalib programs, see below) This was originally designed to imitate a secure attention key	**k**	**t**	**k**	**e**
Shows a stack backtrace for all active CPUs.	**l**	**n**	**l**	**i**
Output current memory information to the console	**m**	**m**	**,**	**m**
Reset the nice level of all high-priority and real-time tasks	**n**	**b**	**n**	**k**
Shut off the system	**o**	**r**	**o**	**y**
Output the current registers and flags to the console	**p**	**l**	**p**	**;**
Display all active high-resolution timers and clock sources.	**q**	**'**	**a**	**q**
Switch the keyboard from raw mode, the mode used by programs such as X11 and svgalib, to XLATE mode	**r**	**p**	**r**	**p**
Sync all mounted filesystems	**s**	**o**	**s**	**r**
Output a list of current tasks and their information to the console	**t**	**y**	**t**	**g**
Remount all mounted filesystems in read-only mode	**u**	**g**	**u**	**l**
Forcefully restores framebuffer console, except for ARM processors, where this key causes ETM buffer dump	**v**	**k**	**v**	**v**
Display list of blocked (D state) tasks	**w**	**,**	**z**	**w**
Used by xmon interface on PPC/PowerPC platforms.	**x**	**q**	**x**	**x**
Show global CPU registers (SPARC-64 specific)	**y**	**f**	**y**	**j**
Dump the ftrace buffer	**z**	**;**	**w**	**z**
Print a summary of available magic SysRq keys	**space**	**space**	**space**	**space**

Uses

A common use of the magic SysRq key is to perform a safe reboot of a Linux computer which has otherwise locked up (abbr. **REISUB**). This can prevent a fsck being required on reboot and gives some programs a chance to save emergency backups of unsaved work.[5] The QWERTY (or AZERTY) mnemonics: "**R**aising **E**lephants **I**s **S**o **U**tterly **B**oring", "**R**eboot **E**ven **I**f **S**ystem **U**tterly **B**roken" or simply the word "BUSIER" read backwards, are often used to remember the following SysRq-keys sequence:

```
unRaw      (take control of keyboard back from X),
tErminate (send SIGTERM to all processes, allowing them to terminate gracefully),
kIll      (send SIGKILL to all processes, forcing them to terminate immediately),
 Sync     (flush data to disk),
 Unmount  (remount all filesystems read-only),
reBoot.
```

When magic SysRq keys are used to kill a frozen graphical program, the program has no chance to restore text mode. This can make everything unreadable. The commands **textmode** (part of SVGAlib) and the **reset** command can restore text mode and make the console readable again.

On distributions that do not include a **textmode** executable, the key command Ctrl + Alt + F1 may sometimes be able to force a return to a text console. (Use F1, F2, F3,..., F(*n*), where *n* is the highest number of text consoles set up by the distribution. Ctrl + Alt + F(*n*+1) would normally be used to reenter GUI mode on a system on which the X server has not crashed.)

Configuration

The feature is controlled both by a compile-time option in the kernel configuration, CONFIG_MAGIC_SYSRQ, and a sysctl kernel parameter, kernel.sysrq.

On newer kernels (since 2.6.12[6]), it is possible to have more fine-grained control over how the magic SysRq key can be used.[7] On these machines, the number written to `/proc/sys/kernel/sysrq` can be zero, one, or a number greater than one which is a bitmask indicating which features to allow.

Alternate ways to invoke Magic SysRq

While the magic SysRq key was originally implemented as part of the kernel's keyboard handler for debugging, the functionality has been also exposed via the proc filesystem and is commonly used to provide extended management capabilities to headless and remote systems.

The Linux daemons `sysrqd`[8] and `tcpconsole`[9] provide a method of accessing SysRq features over a TCP connection after authenticating with a plain-text password. The `hangwatch`[10] daemon will invoke pre-configured sysrq triggers when system load average exceeds a certain threshold.

The Xen hypervisor has functionality to send magic commands to hosted domains via its "xm sysrq" command.[11] Additionally, a sysrq command can be invoked from a Xen paravirtual console by sending a break sequence Ctrl + O followed by the desired key.

Many embedded systems have no attached keyboard, but instead use a serial console for text input/output to the running system. It is possible to invoke a Magic SysRq feature over a serial console by sending a 'break' serial command, followed by the desired key. The method of sending a break is dependent on the terminal program or hardware used to connect to the serial console.

Chromebooks have a keyboard but no dedicated SysRq key. They use Alt + VolumeUp (F10) instead[12][13]

IBM Power servers can invoke the Magic SysRq feature using Ctrl + O followed by the desired key from HMC console.

IBM mainframe partitions can invoke the Magic SysRq feature using ^ + . followed by the desired key on 3270 or HMC console.

See also

- Stop-A, key sequence used to access Sun Microsystems's Open Firmware (OpenBoot)
- Console server
- KVM switch
- System console

References

1. "Linux Magic System Request Key Hacks" (https://www.kernel.org/doc/html/latest/admin-guide/sysrq.html). kernel.org. 2013-08-12. Retrieved 2017-05-21.
2. Khamlichi, Mel (18 Oct 2010), *Use Magic SysRq Combination Key to Recover from a Frozen System in Linux* (http://www.unixmen.com/use-magic-sysrq-combination-key-to-recover-from-a-frozen-system-in-linux-ubuntu/), Unixmen, retrieved 21 Sep 2014.
3. Turner, Glen; Komarinski, Mark F (2003-03-31). "Magic SysRq key" (http://www.tldp.org/HOWTO/Remote-Serial-Console-HOWTO/security-sysrq.html). *Remote Serial Console How-To*. The Linux Documentation Project. Retrieved 2015-05-08.
4. Barnes, Jesse (2008-11-07). "DRM: i915: add mode setting support" (https://git.kernel.org/cgit/linux/kernel/git/torvalds/linux.git/commit/?id=79e539453b34e35f39299a899d263b0a1f1670bd). *Git* (commit). Kernel. Retrieved 2013-08-31.
5. Peddicord, Jacob (2007-09-08). "Fix a Frozen System with the Magic SysRq Keys" (http://fosswire.com/post/2007/09/fix-a-frozen-system-with-the-magic-sysrq-keys/). Fosswire. Retrieved 2013-08-31.
6. Calleja, Diego (2012-07-17). "Linux 2.6.12" (http://kernelnewbies.org/Linux_2_6_12). Linux Kernel Newbies. Retrieved 2013-08-31.
7. "QA/Sysrq" (https://fedoraproject.org/wiki/QA/Sysrq). Fedora Project. 2013-05-09. Retrieved 2015-04-09.
8. Danjou, Julien. "sysrqd" (https://github.com/jd/sysrqd) (project). Retrieved 2016-09-13.
9. van Heusden, Folkert. "tcpconsole" (http://www.vanheusden.com/tcpconsole/). Retrieved 2013-08-31.
10. Morgan, Paul. "hangwatch" (https://github.com/jumanjiman/hangwatch). GitHub. Retrieved 2013-08-31.
11. "27. The xm command quick reference", *Enterprise Linux 5 Virtualization Guide* (https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux/5/html/Virtualization/chap-Virtualization-The_xm_command_quick_reference.html), RedHat, retrieved 2013-08-31
12. "Change 29110: CHROMIUM: sysrq: treat F10 as magic sysrq key" (https://chromium-review.googlesource.com/#/c/29110/2/drivers/tty/sysrq.c). Retrieved 2016-07-11.
13. "Chromium OS dev: SysRq in ChromiumOS' kernel" (https://groups.google.com/a/chromium.org/d/msg/chromium-os-dev/HnW8maLcDas/NJU1HFmE76kJ). Retrieved 2016-07-11.

External links

- "Documentation/sysrq.txt" (https://git.kernel.org/cgit/linux/kernel/git/torvalds/linux.git/tree/Documentation/admin-guide/sysrq.rst). *Linux* (kernel source tree). kernel.org. Retrieved 2017-01-31.
- Morton, Andrew (2013-03-13). "Kernel patch to the -mm tree that added the /proc/sysrq-trigger feature: This makes sysrq facilities available to remote users" (https://www.kernel.org/pub/linux/kernel/people/akpm/patches/2.5/2.5.64/2.5.64-mm7/broken-out/proc-sysrq-trigger.patch). Retrieved 2015-04-09.
- Lawrence, Tony (2005-04-13). "Magic Sysrq Key on Linux" (http://aplawrence.com/Words2005/2005_04_13.html). *AP Lawrence*. Retrieved 2015-04-09.
- Gall, Tom (2000-04-01). "Magic sys request" (https://web.archive.org/web/20070527215139/http://www.ibm.com/developerworks/linux/library/l-magic.html). *Developer works*. IBM. Archived from the original (http://www.ibm.com/developerworks/linux/library/l-magic.html) on 2007-05-27. Retrieved 2013-09-03.