1 Linux Magic System Request Key Hacks

Documentation for sysrq.c

1.1 What are the 'command' keys?

Command	Function
`b`	Will immediately reboot the system without syncing or unmounting your disks.
`c`	Will perform a system crash by a NULL pointer dereference. A crashdump will be taken if configured.
`d`	Shows all locks that are held.
`e`	Send a SIGTERM to all processes, except for init.
`f`	Will call the oom killer to kill a memory hog process, but do not
pani	c if nothing can be killed.
`g`	Used by kgdb (kernel debugger)
`h`	Will display help (actually any other key than those listed here will display help. but `h` is easy to remember :-)
`i`	Send a SIGKILL to all processes, except for init.
`j`	Forcibly "Just thaw it" - filesystems frozen by the FIFREEZE ioctl.
`k`	Secure Access Key (SAK) Kills all programs on the current virtual console. NOTE: See important comments below in SAK section.
`l`	Shows a stack backtrace for all active CPUs.
`m`	Will dump current memory info to your console.
`n`	Used to make RT tasks nice-able
`o`	Will shut your system off (if configured and supported).
`p`	Will dump the current registers and flags to your console.
`q`	Will dump per CPU lists of all armed hrtimers (but NOT regular timer_list timers) and detailed information about all clockevent devices.
`r`	Turns off keyboard raw mode and sets it to XLATE.
`s`	Will attempt to sync all mounted filesystems.
`t`	Will dump a list of current tasks and their information to your console.
`u`	Will attempt to remount all mounted filesystems read-only.
`v`	Forcefully restores framebuffer console
`v`	Causes ETM buffer dump [ARM-specific]
`w`	Dumps tasks that are in uninterruptable (blocked) state.
`x`	Used by xmon interface on ppc/powerpc platforms. Show global PMU Registers on sparc64. Dump all TLB entries on MIPS.
`y`	Show global CPU Registers [SPARC-64 specific]
`z`	Dump the ftrace buffer
`0-9`	Sets the console log level, controlling which kernel messages will be printed to your console. (`0`, for example would make it so that only emergency messages like PANICs or OOPSes would make it to your console.)

1.2 Okay, so what can I use them for?

- Well, `unraw(r)` is very handy when your X server or a svgalib program crashes.
- `sak(k)` (Secure Access Key) is useful when you want to be sure there is no trojan program running at console which could grab your password when you would try to login. It will kill all programs on given console, thus letting you make sure that the login prompt you see is actually the one from init, not some trojan program.

  In its true form it is not a true SAK like the one in a c2 compliant system, and it should not be mistaken as such.

  It seems others find it useful as (System Attention Key) which is useful when you want to exit a program that will not let you switch consoles. (For example, X or a svgalib program.)
- `reboot(b)` is good when you're unable to shut down. But you should also `sync(s)` and `umount(u)` first.
- `crash(c)` can be used to manually trigger a crashdump when the system is hung. Note that this just triggers a crash if there is no dump mechanism available.
- `sync(s)` is great when your system is locked up, it allows you to sync your disks and will certainly lessen the chance of data loss and fscking. Note that the sync hasn't taken place until you see the "OK" and "Done" appear on the screen. (If the kernel is really in strife, you may not ever get the OK or Done message...)
- `umount(u)` is basically useful in the same ways as `sync(s)`. I generally `sync(s)`, `umount(u)`, then `reboot(b)` when my system locks. It's saved me many a fsck. Again, the unmount (remount read-only) hasn't taken place until you see the "OK" and "Done" message appear on the screen.
- The loglevels `0-9` are useful when your console is being flooded with kernel messages you do not want to see. Selecting `0` will prevent all but the most urgent kernel messages from reaching your console. (They will still be logged if syslogd/klogd are alive, though.)
- `term(e)` and `kill(i)` are useful if you have some sort of runaway process you are unable to kill any other way, especially if it's spawning other processes.
- "just thaw `it(j)`" is useful if your system becomes unresponsive due to a frozen (probably root) filesystem via the FIFREEZE ioctl.

1.3 Credits

Written by Mydraal <`vulpyne@vulpyne.net`> Updated by Adam Sulmicki <`adam@cfar.umd.edu`> Updated by Jeremy M. Dolan <`jmd@turbogeek.org`> 2001/01/28 10:15:59 Added to by Crutcher Dunnavant <`crutcher+kernel@datastacks.com`>