RHEL: What is "SysRq key" and how to use it

Article Number: 141 | Rating: Unrated | Last Updated: Sat, Jun 2, 2018 9:10 AM

RHEL: What is "SysRq key" and how to use it

```
# Tested on RHEL 5, 6 & 7


# What is the "Magic" 'SysRq' key?
# -------------------------------------------------------------------------------
---------

# According to the Linux kernel documentation:
#
#  It is a 'magical' key combo you can hit which the kernel will respond to
regardless of
#  whatever else it is doing, even if the console is unresponsive.
#
# The 'SysRq' key is one of the best (and sometimes the only) way to determine what
a
# machine is really doing. It is useful when a system appears to be "hung" or for
# diagnosing elusive, transient, kernel-related problems.



# How do I enable and disable the 'SysRq' key?
# -------------------------------------------------------------------------------
---------

# For security reasons, Red Hat Enterprise Linux disables the 'SysRq' key by
default. To
# enable it, run:

echo 1 > /proc/sys/kernel/sysrq

# Or: sysctl -w kernel.sysrq=1

# List of possible values in /proc/sys/kernel/sysrq:
```

```
#
#  0 - disable sysrq completely
#  1 - enable all functions of sysrq
#    >1 - bitmask of allowed sysrq functions (see below for detailed function
description):
#      2 - enable control of console logging level
#      4 - enable control of keyboard (SAK, unraw)
#      8 - enable debugging dumps of processes etc.
#     16 - enable sync command
#     32 - enable remount read-only
#     64 - enable signalling of processes (term, kill, oom-kill)
#    128 - allow reboot/poweroff
#    256 - allow nicing of all RT tasks


# To disable it:

echo 0 > /proc/sys/kernel/sysrq

# Or: sysctl -w kernel.sysrq=0


# To enable it permanently, set the kernel.sysrq value to 1. That will cause it to
be
# enabled on start up

# RHEL 5 & 6
vi /etc/sysctl.conf
   kernel.sysrq = 1

# RHEL 7
vi /usr/lib/sysctl.d/50-default.conf
   kernel.sysrq = 1


# Since enabling 'SysRq' gives you physical console access extra abilities, it is
recommended
# to disable it when not troubleshooting a problem or to ensure that physical
console
# access is properly secured.



# How do I trigger a 'SysRq' event?
# ------------------------------------------------------------------------------
---------
```

```
# There are several ways to trigger a 'SysRq' event. On a normal system, with an AT
# keyboard, events can be triggered from the console with the following key
combination:

# Alt+PrintScreen+[CommandKey]

# For instance, to tell the kernel to dump memory info (command key "m"), you would
hold
# down the "Alt" and "Print Screen keys", and then hit the m key.

# Note that this will not work from an X Window System screen. You should first
change to
# a text virtual terminal. Hit Ctrl+Alt+F1 to switch to the first virtual console
prior to
# hitting the 'SysRq' key combination.

# On a serial console, you can achieve the same effect by sending a Break signal to
the
# console and then hitting the command key within 5 seconds. This also works for
virtual
# serial console access through an out-of-band service processor or remote console
like
# HP iLO, Sun ILOM and IBM RSA.
# Refer to service processor specific documentation for details on how to send a
Break
# signal; for example, How to trigger SysRq over an HP iLo Virtual Serial Port
(VSP).

# If you have a root shell on the machine (and the system is responding enough for
you to
# do so), you can also write the command key character to the /proc/sysrq-trigger
file.
# This is useful for triggering this info when you are not on the system console or
for
# triggering it from scripts.

echo 'm' > /proc/sysrq-trigger



# When I trigger a 'SysRq' event that generates output, where does it go?
# ------------------------------------------------------------------------------
---------

# When a 'SysRq' command is triggered, the kernel will print out the information to
```

```
the
# kernel ring buffer and to the system console. This information is normally logged
via
# syslog to /var/log/messages.

# Unfortunately, when dealing with machines that are extremely unresponsive,
syslogd is
# often unable to log these events. In these situations, provisioning a serial
console is
# often recommended for collecting the data.



# What sort of 'SysRq' events can be triggered?
# ------------------------------------------------------------------------------
---------

# There are several 'SysRq' events that can be triggered once the 'SysRq' facility
is
# enabled. These vary somewhat between kernel versions, but there are a few that
are
# commonly used:

#  m - dump information about memory allocation
#  t - dump thread state information
#  p - dump current CPU registers and flags
#  c - intentionally crash the system (useful for forcing a disk or netdump)
#  s - immediately sync all mounted filesystems
#  u - immediately remount all filesystems read-only
#  b - immediately reboot the machine
#  o - immediately power off the machine (if configured and supported)
#  f - start the Out Of Memory Killer (OOM)
#  w - dumps tasks that are in uninterruptible (blocked) state
```