Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'makedumpfile.8' command

$ man makedumpfile.8

MAKEDUMPFILE(8) Linux System Administrator's Manual MAKEDUMPFILE(8)

NAME

makedumpfile - make a small dumpfile of kdump

SYNOPSIS

makedumpfile [OPTION] [-x VMLINUX|-i VMCOREINFO] VMCORE DUMPFILE

makedumpfile -F [OPTION] [-x VMLINUX|-i VMCOREINFO] VMCORE

makedumpfile [OPTION] -x VMLINUX [--config FILTERCONFIGFILE] [--eppic

EPPICMACRO] VMCORE DUMPFILE

makedumpfile -R DUMPFILE

makedumpfile --split [OPTION] [-x VMLINUX|-i VMCOREINFO] VMCORE DUMP?

FILE1 DUMPFILE2 [DUMPFILE3 ..]

makedumpfile [OPTION] [-x VMLINUX|-i VMCOREINFO] --num-threads THREAD?

NUM VMCORE DUMPFILE

makedumpfile --reassemble DUMPFILE1 DUMPFILE2 [DUMPFILE3 ..] DUMPFILE

makedumpfile -g VMCOREINFO -x VMLINUX

makedumpfile [OPTION] [--xen-syms XEN-SYMS|--xen-vmcoreinfo VMCORE?

INFO] VMCORE DUMPFILE

makedumpfile --dump-dmesg [--partial-dmesg] [-x VMLINUX|-i VMCOREINFO]

VMCORE LOGFILE

makedumpfile [OPTION] -x VMLINUX --diskset=VMCORE1 --diskset=VMCORE2

[--diskset=VMCORE3 ..] DUMPFILE

makedumpfile -h

makedumpfile -v

DESCRIPTION

With kdump, the memory image of the first kernel (called "panicked ker?
nel") can be taken as /proc/vmcore while the second kernel (called
"kdump kernel" or "capture kernel") is running. This document repre?
sents /proc/vmcore as VMCORE. makedumpfile makes a small DUMPFILE by
compressing dump data or by excluding unnecessary pages for analysis,
or both. makedumpfile needs the first kernel's debug information, so
that it can distinguish unnecessary pages by analyzing how the first
kernel uses the memory. The information can be taken from VMLINUX or
VMCOREINFO.

makedumpfile can exclude the following types of pages while copying VM?
CORE to DUMPFILE, and a user can choose which type of pages will be ex?
cluded.

- Pages filled with zero
- Cache pages without private flag (non-private cache)
- Cache pages with private flag (private cache)
- User process data pages
- Free pages

makedumpfile provides two DUMPFILE formats (the ELF format and the
kdump-compressed format). By default, makedumpfile makes a DUMPFILE in
the kdump-compressed format. The kdump-compressed format is readable
only with the crash utility, and it can be smaller than the ELF format
because of the compression support. The ELF format is readable with GDB
and the crash utility. If a user wants to use GDB, DUMPFILE format has
to be explicitly specified to be the ELF format.

Apart from the exclusion of unnecessary pages mentioned above, make?
dumpfile allows user to filter out targeted kernel data. The filter
config file can be used to specify kernel/module symbols and its mem?
bers that need to be filtered out through the erase command syntax.
makedumpfile reads the filter config and builds the list of memory ad?
dresses and its sizes after processing filter commands. The memory lo?
cations that require to be filtered out are then poisoned with charac?
ter 'X' (58 in Hex). Refer to makedumpfile.conf(5) for file format.

Eppic macros can also be used to specify kernel symbols and its members that need to be filtered. Eppic provides C semantics including language constructs such as conditional statements, logical and arithmetic oper? ators, functions, nested loops to traverse and erase kernel data. --ep? pic requires eppic_makedumpfile.so and eppic library. eppic_makedump? file.so can be built from makedumpfile source. Refer to http://code.google.com/p/eppic/ to build eppic library libeppic.a and for more information on writing eppic macros.

To analyze the first kernel's memory usage, makedumpfile can refer to VMCOREINFO instead of VMLINUX. VMCOREINFO contains the first kernel's information (structure size, field offset, etc.), and VMCOREINFO is small enough to be included into the second kernel's initrd.

If the second kernel is running on its initrd without mounting a root file system, makedumpfile cannot refer to VMLINUX because the second kernel's initrd cannot include a large file like VMLINUX. To solve the problem, makedumpfile makes VMCOREINFO beforehand, and it refers to VM? COREINFO instead of VMLINUX while the second kernel is running.

VMCORE has contained VMCOREINFO since linux-2.6.24, and a user does not need to specify neither -x nor -i option.

If the second kernel is running on its initrd without mounting any file system, a user needs to transport the dump data to a remote host. To transport the dump data by SSH, makedumpfile outputs the dump data in the intermediate format (the flattened format) to the standard output. By piping the output data to SSH, a user can transport the dump data to a remote host. Note that analysis tools (crash utility before version 5.1.2 or GDB) cannot read the flattened format directly, so on a remote host the received data in the flattened format needs to be rearranged to a readable DUMPFILE format by makedumpfile (or makedumpfile-R.pl).

makedumpfile can read a DUMPFILE in the kdump-compressed format instead of VMCORE and re-filter it. This feature is useful in situation that users need to reduce the file size of DUMPFILE for sending it somewhere by ftp/scp/etc. (If all of the page types, which are specified by a new dump_level, are excluded from an original DUMPFILE already, a new DUMP?

FILE is the same as an original DUMPFILE.)

For example, makedumpfile can create a DUMPFILE of dump_level 31 from the one of dump_level 3 like the following:

Example:

makedumpfile -c -d 3 /proc/vmcore dumpfile.1

makedumpfile -c -d 31 dumpfile.1 dumpfile.2

makedumpfile can read VMCORE(s) in three kinds of sadump formats: single partition format, diskset format and media backup format, and can convert each of them into kdump-compressed format with filtering and compression processing. Note that for VMCORE(s) created by sadump, you always need to pass VMLINUX with -x option. Also, to pass multiple VMCOREs created on diskset configuration, you need to use --diskset option.

OPTIONS

-c,-l,-p,-z

Compress dump data by the page using the following compression library respectively:

-c : zlib

-l : lzo

-p : snappy

-z : zstd

(-l, -p and -z option need USELZO=on, USESNAPPY=on and USEZSTD=on respectively when building makedumpfile)

A user cannot specify this option with -E option, because the ELF format does not support compressed data.

Example:

makedumpfile -c -d 31 -x vmlinux /proc/vmcore dumpfile

-d dump_level

Specify the type of unnecessary page for analysis.

Pages of the specified type are not copied to DUMPFILE. The page type marked in the following table is excluded. A user can specify multiple page types by setting the sum of each page type for dump_level. The maximum of dump_level is 31. Note that a

dump_level for Xen dump filtering is 0 or 1 on a machine other than x86_64. On a x86_64 machine, even 2 or bigger dump level will be effective if you specify domain-0's vmlinux with -x op? tion. Then the pages are excluded only from domain-0.

If specifying multiple dump_levels with the delimiter ',', make? dumpfile retries to create DUMPFILE using the next dump_level when the size of a dumpfile exceeds the limit specified with '-L' or when a "No space on device" error happens. For example, if dump_level is "11,31" and makedumpfile fails with dump_level 11, makedumpfile retries with dump_level 31.

Example:

makedumpfile -d 11 -x vmlinux /proc/vmcore dumpfile

makedumpfile -d 11,31 -x vmlinux /proc/vmcore dumpfile

Base level:

dump_level consists of five bits, so there are five base levels to specify the type of unnecessary page.

1 : Exclude the pages filled with zero.

2 : Exclude the non-private cache pages.

4 : Exclude all cache pages.

8 : Exclude the user process data pages.

16 : Exclude the free pages.

Here is the all combinations of the bits.

		non-		
dump level	zero page	private cache	private cache	user data	free page
0					
1	X				
2		X			
3	X	X			
4		X	X		
5	X	X	X		
6		X	X		

7 | X | X | X | | |

8 | | | | | X |

9 | X | | | | X |

10 | | X | | | X |

11 | X | X | | | X |

12 | | X | X | | X |

13 | X | X | X | | X |

14 | | X | X | | X |

15 | X | X | X | | X |

16 | | | | | | X

17 | X | | | | | X

18 | | X | | | | X

19 | X | X | | | | X

20 | | X | X | | | X

21 | X | X | X | | | X

22 | | X | X | | | X

23 | X | X | X | | | X

24 | | | | | X | X

25 | X | | | | X | X

26 | | X | | | X | X

27 | X | X | | | X | X

28 | | X | X | | X | X

29 | X | X | X | | X | X

30 | | X | X | | X | X

31 | X | X | X | | X | X

-L SIZE

Limit the size of the output file to SIZE bytes. An incomplete DUMPFILE or LOGFILE is written if the size would otherwise ex? ceed SIZE.

-E Create DUMPFILE in the ELF format.

This option cannot be specified with the -c, -l or -p options, because the ELF format does not support compressed data.

Example:

makedumpfile -E -d 31 -x vmlinux /proc/vmcore dumpfile

-f Force existing DUMPFILE to be overwritten and mem-usage to work with older kernel as well.

Example:

makedumpfile -f -d 31 -x vmlinux /proc/vmcore dumpfile

This command overwrites DUMPFILE even if it already exists.

makedumpfile -f --mem-usage /proc/kcore

Kernel version lesser than v4.11 will not work with --mem-usage functionality until it has been patched with upstream commit 464920104bf7. Therefore if you have patched your older kernel then use -f.

-x VMLINUX

Specify the first kernel's VMLINUX with debug information to an? alyze the first kernel's memory usage.

This option is necessary if VMCORE does not contain VMCOREINFO, [-i VMCOREINFO] is not specified, and dump_level is 2 or more.

The page size of the first kernel and the second kernel should match.

Example:

makedumpfile -d 31 -x vmlinux /proc/vmcore dumpfile

-i VMCOREINFO

Specify VMCOREINFO instead of VMLINUX for analyzing the first kernel's memory usage.

VMCOREINFO should be made beforehand by makedumpfile with -g op? tion, and it contains the first kernel's information.

This option is necessary if VMCORE does not contain VMCOREINFO, [-x VMLINUX] is not specified, and dump_level is 2 or more.

Example:

makedumpfile -d 31 -i vmcoreinfo /proc/vmcore dumpfile

-g VMCOREINFO

Generate VMCOREINFO from the first kernel's VMLINUX with debug information.

VMCOREINFO must be generated on the system that is running the

first kernel. With -i option, a user can specify VMCOREINFO gen?
erated on the other system that is running the same first ker?
nel. [-x VMLINUX] must be specified.

Example:

makedumpfile -g vmcoreinfo -x vmlinux

--config FILTERCONFIGFILE

Used in conjunction with -x VMLINUX option, to specify the fil?
ter config file FILTERCONFIGFILE that contains erase commands to
filter out desired kernel data from vmcore while creating DUMP?
FILE. For filter command syntax please refer to makedump?
file.conf(5).

--eppic EPPICMACRO

Used in conjunction with -x VMLINUX option, to specify the eppic
macro file that contains filter rules or directory that contains
eppic macro files to filter out desired kernel data from vmcore
while creating DUMPFILE. When directory is specified, all the
eppic macros in the directory are processed.

-F Output the dump data in the flattened format to the standard
output for transporting the dump data by SSH.

Analysis tools (crash utility before version 5.1.2 or GDB) can?
not read the flattened format directly. For analysis, the dump
data in the flattened format should be rearranged to a normal
DUMPFILE (readable with analysis tools) by -R option. By which
option is specified with -F option, the format of the rearranged
DUMPFILE is fixed. In other words, it is impossible to specify
the DUMPFILE format when the dump data is rearranged with -R op?
tion. If specifying -E option with -F option, the format of the
rearranged DUMPFILE is the ELF format. Otherwise, it is the
kdump-compressed format. All the messages are output to standard
error output by -F option because standard output is used for
the dump data.

Example:

makedumpfile -F -c -d 31 -x vmlinux /proc/vmcore \

| ssh user@host "cat > dumpfile.tmp"

makedumpfile -F -c -d 31 -x vmlinux /proc/vmcore \

| ssh user@host "makedumpfile -R dumpfile"

makedumpfile -F -E -d 31 -i vmcoreinfo /proc/vmcore \

| ssh user@host "makedumpfile -R dumpfile"

makedumpfile -F -E --xen-vmcoreinfo VMCOREINFO /proc/vmcore \

| ssh user@host "makedumpfile -R dumpfile"

-R Rearrange the dump data in the flattened format from the stan?

dard input to a normal DUMPFILE (readable with analysis tools).

Example:

makedumpfile -R dumpfile < dumpfile.tmp

makedumpfile -F -d 31 -x vmlinux /proc/vmcore \

| ssh user@host "makedumpfile -R dumpfile"

Instead of using -R option, a perl script "makedumpfile-R.pl"

rearranges the dump data in the flattened format to a normal

DUMPFILE, too. The perl script does not depend on architecture,

and most systems have perl command. Even if a remote host does

not have makedumpfile, it is possible to rearrange the dump data

in the flattened format to a readable DUMPFILE on a remote host

by running this script.

Example:

makedumpfile -F -d 31 -x vmlinux /proc/vmcore \

| ssh user@host "makedumpfile-R.pl dumpfile"

--split

Split the dump data to multiple DUMPFILEs in parallel. If speci?

fying DUMPFILEs on different storage devices, a device can share

I/O load with other devices and it reduces time for saving the

dump data. The file size of each DUMPFILE is smaller than the

system memory size which is divided by the number of DUMPFILEs.

This feature supports only the kdump-compressed format.

Example:

makedumpfile --split -d 31 -x vmlinux /proc/vmcore dumpfile1

dumpfile2

--num-threads THREADNUM

Using multiple threads to read and compress data of each page in parallel. And it will reduces time for saving DUMPFILE. Note that if the usable cpu number is less than the thread number, it may lead to great performance degradation. This feature only supports creating DUMPFILE in kdump-comressed format from VMCORE in kdump-compressed format or elf format.

Example:

makedumpfile -d 31 --num-threads 4 /proc/vmcore dumpfile

--reassemble

Reassemble multiple DUMPFILEs, which are created by --split op? tion, into one DUMPFILE. dumpfile1 and dumpfile2 are reassembled into dumpfile on the following example.

Example:

makedumpfile --reassemble dumpfile1 dumpfile2 dumpfile

-b <order>

Cache 2^order pages in ram when generating DUMPFILE before writ? ing to output. The default value is 4.

--cyclic-buffer buffer_size

Specify the buffer size in kilo bytes for bitmap data. Filter? ing processing will be divided into multi cycles to fix the mem? ory consumption, the number of cycles is represented as:

num_of_cycles = system_memory / (buffer_size * 1024 * bit_per_bytes * page_size)

The lesser number of cycles, the faster working speed is ex? pected. By default, buffer_size will be calculated automati? cally depending on system memory size, so ordinary users don't need to specify this option.

Example:

makedumpfile --cyclic-buffer 1024 -d 31 -x vmlinux /proc/vm? core dumpfile

--splitblock-size splitblock_size

Specify the splitblock size in kilo bytes for analysis with

--split. If --splitblock N is specified, difference of each splitted dumpfile size is at most N kilo bytes.

Example:

makedumpfile --splitblock-size 1024 -d 31 -x vmlinux --split /proc/vmcore dumpfile1 dumpfile2

--work-dir

Specify the working directory for the temporary bitmap file. If this option isn't specified, the bitmap will be saved on memory. Filtering processing has to do 2 pass scanning to fix the memory consumption, but it can be avoided by using working directory on file system. So if you specify this option, the filtering speed may be bit faster.

Example:

makedumpfile --work-dir /tmp -d 31 -x vmlinux /proc/vmcore dumpfile

--non-mmap

Never use mmap(2) to read VMCORE even if it supports mmap(2). Generally, reading VMCORE with mmap(2) is faster than without it, so ordinary users don't need to specify this option. This option is mainly for debugging.

Example:

makedumpfile --non-mmap -d 31 -x vmlinux /proc/vmcore dumpfile

--xen-syms XEN-SYMS

Specify the XEN-SYMS with debug information to analyze the xen's memory usage. This option extracts the part of xen and do? main-0.

Example:

makedumpfile -E --xen-syms xen-syms /proc/vmcore dumpfile

--xen-vmcoreinfo VMCOREINFO

Specify VMCOREINFO instead of XEN-SYMS for analyzing the xen's memory usage.

VMCOREINFO should be made beforehand by makedumpfile with -g op? tion, and it contains the xen's information.

Example:

makedumpfile -E --xen-vmcoreinfo VMCOREINFO /proc/vmcore dump?

file

-X Exclude all the user domain pages from Xen kdump's VMCORE, and

extracts the part of xen and domain-0. If VMCORE contains VMCOR?

EINFO for Xen, it is not necessary to specify --xen-syms and

--xen-vmcoreinfo.

Example:

makedumpfile -E -X /proc/vmcore dumpfile

--xen_phys_start xen_phys_start_address

This option is only for x86_64. Specify the xen_phys_start_ad?

dress, if the xen code/data is relocatable and VMCORE does not

contain xen_phys_start_address in the CRASHINFO.

xen_phys_start_address can be taken from the line of "Hypervisor

code and data" in /proc/iomem. For example, specify 0xcee00000

as xen_phys_start_address if /proc/iomem is the following:

```
----------------------------------------------------
# cat /proc/iomem
...
  cee00000-cfd99999 : Hypervisor code and data
...
----------------------------------------------------
```

Example:

makedumpfile -E -X --xen_phys_start 0xcee00000 /proc/vmcore

dumpfile

--message-level message_level

Specify the message types.

Users can restrict outputs printed by specifying message_level

with this option. The message type marked with an X in the fol?

lowing table is printed. For example, according to the table,

specifying 7 as message_level means progress indicator, common

message, and error message are printed, and this is a default

value. Note that the maximum value of message_level is 31.

```
 message | progress | common  | error   | debug   | report
 level   | indicator| message | message | message | message
---------+----------+---------+---------+---------+---------
       0 |          |         |         |         |
       1 |    X     |         |         |         |
       2 |          |    X    |         |         |
       3 |    X     |    X    |         |         |
       4 |          |         |    X    |         |
       5 |    X     |         |    X    |         |
       6 |          |    X    |    X    |         |
     * 7 |    X     |    X    |    X    |         |
       8 |          |         |         |    X    |
       9 |    X     |         |         |    X    |
      10 |          |    X    |         |    X    |
      11 |    X     |    X    |         |    X    |
      12 |          |         |    X    |    X    |
      13 |    X     |         |    X    |    X    |
      14 |          |    X    |    X    |    X    |
      15 |    X     |    X    |    X    |    X    |
      16 |          |         |         |         |    X
      17 |    X     |         |         |         |    X
      18 |          |    X    |         |         |    X
      19 |    X     |    X    |         |         |    X
      20 |          |         |    X    |         |    X
      21 |    X     |         |    X    |         |    X
      22 |          |    X    |    X    |         |    X
      23 |    X     |    X    |    X    |         |    X
      24 |          |         |         |    X    |    X
      25 |    X     |         |         |    X    |    X
      26 |          |    X    |         |    X    |    X
      27 |    X     |    X    |         |    X    |    X
      28 |          |         |    X    |    X    |    X
      29 |    X     |         |    X    |    X    |    X
```

30 | | X | X | X | X

31 | X | X | X | X | X

--vtop virtual_address

This option is useful, when user debugs the translation problem of virtual address. If specifing virtual_address, its physical address is printed. It makes debugging easy by comparing the output of this option with the one of "vtop" subcommand of the crash utility. "--vtop" option only prints the translation out? put, and it does not affect the dumpfile creation.

--dump-dmesg

This option overrides the normal behavior of makedumpfile. In? stead of compressing and filtering a VMCORE to make it smaller, it simply extracts the dmesg log from a VMCORE and writes it to the specified LOGFILE. If a VMCORE does not contain VMCOREINFO for dmesg, it is necessary to specfiy [-x VMLINUX] or [-i VMCOR? EINFO].

Example:

makedumpfile --dump-dmesg /proc/vmcore dmesgfile

makedumpfile --dump-dmesg -x vmlinux /proc/vmcore dmesgfile

--partial-dmesg

This option will make --dump-dmesg extract only dmesg logs since that buffer was last cleared on the crashed kernel, through "dmesg --clear" for example.

--mem-usage

This option is currently supported on x86_64, arm64, ppc64 and s390x. This option is used to show the page numbers of current system in different use. It should be executed in 1st kernel. By the help of this, user can know how many pages is dumpable when different dump_level is specified. It analyzes the 'System Ram' and 'kernel text' program segment of /proc/kcore excluding the crashkernel range, then calculates the page number of different kind per vmcoreinfo. So currently /proc/kcore need be specified explicitly.

Example:

makedumpfile --mem-usage /proc/kcore

--diskset=VMCORE

Specify multiple VMCOREs created on sadump diskset configuration the same number of times as the number of VMCOREs in increasing order from left to right. VMCOREs are assembled into a single DUMPFILE.

Example:

makedumpfile -x vmlinux --diskset=vmcore1 --diskset=vmcore2 dumpfile

-D Print debugging message.

-h (--help)

Show help message and LZO/snappy support status (enabled/disabled).

-v Show the version of makedumpfile.

--check-params

Only check whether the command-line parameters are valid or not, and exit. Preferable to be given as the first parameter.

--dry-run

Do not write the output dump file while still performing operations specified by other options. This option cannot be used with the --dump-dmesg, --reassemble and -g options.

--show-stats

Display report messages. This is an alternative to enabling bit 4 in the level provided to --message-level.

ENVIRONMENT VARIABLES

TMPDIR This environment variable is used in 1st kernel environment for a temporary memory bitmap file. If your machine has a lots of memory and you use small tmpfs on /tmp, makedumpfile can fail for a little memory because makedumpfile makes a very large temporary memory bitmap file in this case. To avoid this fail?ure, you should specify --work-dir option to use file system on storage for the bitmap file.

DIAGNOSTICS

makedumpfile exits with the following value.

0 : makedumpfile succeeded.

1 : makedumpfile failed without the following reasons.

2 : makedumpfile failed due to the different version between VMLINUX and VMCORE.

AUTHORS

Written by Masaki Tachibana, and Ken'ichi Ohmichi.

SEE ALSO

crash(8), gdb(1), kexec(8), makedumpfile.conf(5)