# Core dump

From Wikipedia, the free encyclopedia

In computing, a **core dump** (in Unix parlance), **memory dump**, or **system dump**[1] consists of the recorded state of the working memory of a computer program at a specific time, generally when the program has crashed or otherwise terminated abnormally.[2] In practice, other key pieces of program state are usually dumped at the same time, including the processor registers, which may include the program counter and stack pointer, memory management information, and other processor and operating system flags and information. Core dumps are often used to assist in diagnosing and debugging errors in computer programs.

The name comes from magnetic core memory,[3] the principal form of random access memory from the 1950s to the 1970s. The name has remained long after magnetic core technology became obsolete.

On many operating systems, a fatal error in a program automatically triggers a core dump; by extension the phrase "to dump core" has come to mean, in many cases, any fatal error, regardless of whether a record of the program memory results. The term "core dump", "memory dump", or just "dump" has also become a jargon to indicate any storing of a large amount of raw data for further examination or other purposes.[4][5]

## Contents

## Background

Before the advent of disk operating systems and the ability to record large data files, core dumps were paper printouts[6] of the contents of memory, typically arranged in columns of octal or hexadecimal numbers (a "hex dump"), sometimes accompanied by their interpretations as machine language instructions, text strings, or decimal or floating-point numbers (*cf.* disassembler).

Instead of only displaying the contents of the applicable memory, modern operating systems typically generate a file containing an image of the memory belonging to the crashed process, or the memory images of parts of the address space related to that process, along with other information such as the values of processor registers, program counter, system flags, and other information useful in determining the root cause of the crash. These files can be viewed as text, printed, or analysed with specialised tools such as elfdump on Unix and Unix-like systems, objdump and kdump on Linux, WinDbg on Microsoft Windows, Valgrind, or other debuggers.

Modern core dump files and error messages typically use hexadecimal encoding, as decimal and octal representations are less convenient to the programmer.

# Uses

Core dumps can serve as useful debugging aids in several situations. On early standalone or batch-processing systems, core dumps allowed a user to debug a program without monopolizing the (very expensive) computing facility for debugging; a printout could also be more convenient than debugging using switches and lights.

On shared computers, whether time-sharing, batch processing, or server systems, core dumps allow off-line debugging of the operating system, so that the system can go back into operation immediately.

Core dumps allow a user to save a crash for later or off-site analysis, or comparison with other crashes. For embedded computers, it may be impractical to support debugging on the computer itself, so analysis of a dump may take place on a different computer. Some operating systems such as early versions of Unix did not support attaching debuggers to running processes, so core dumps were necessary to run a debugger on a process's memory contents.

Core dumps can be used to capture data freed during dynamic memory allocation and may thus be used to retrieve information from a program that is no longer running. In the absence of an interactive debugger, the core dump may be used by an assiduous programmer to determine the error from direct examination.

# Analysis

A core dump represents the complete contents of the dumped regions of the address space of the dumped process. Depending on the operating system, the dump may contain few or no data structures to aid interpretation of the memory regions. In these systems, successful interpretation requires that the program or user trying to interpret the dump understands the structure of the program's memory use.

A debugger can use a symbol table, if one exists, to help the programmer interpret dumps, identifying variables symbolically and displaying source code; if the symbol table is not available, less interpretation of the dump is possible, but there might still be enough possible to determine the cause of the problem. There are also special-purpose tools called dump analyzers to analyze dumps. One popular tool, available on many operating systems, is the GNU binutils' objdump.

On modern Unix-like operating systems, administrators and programmers can read core dump files using the GNU Binutils Binary File Descriptor library (BFD), and the GNU Debugger (gdb) and objdump that use this library. This library will supply the raw data for a given address in a memory region from a core dump; it does not know anything about variables or data structures in that memory region, so the application using the library to read the core dump will have to determine the addresses of variables and determine the layout of data structures itself, for example by using the symbol table for the program undergoing debugging.

Analysts of crash dumps from Linux systems can use kdump or the Linux Kernel Crash Dump (LKCD).[7]

Core dumps can save the context (state) of a process at a given state for returning to it later. Systems can be made highly available by transferring core between processors, sometimes via core dump files themselves.

Core can also be dumped onto a remote host over a network (which is a security risk).[8]

# Core dump files

## Format

In older and simpler operating systems, each process had a contiguous address-space, so a core dump file was simply a binary file with the sequence of bytes or words. In modern operating systems, a process address space may have gaps, and share pages with other processes or files, so more elaborate representations are used; they may also include other information about the state of the program at the time of the dump.

In Unix-like systems, core dumps generally use the standard executable image-format:

- a.out in older versions of Unix,
- ELF in modern Linux, System V, Solaris, and BSD systems,
- Mach-O in macOS, *etc.*

### Naming

- Since Solaris 8, system utility `coreadm` allows the name and location of core files to be configured.
- Dumps of user processes are traditionally created as `core`. On Linux (since versions 2.4.21 and 2.6 of the Linux kernel mainline), a different name can be specified via procfs using the `/proc/sys/kernel/core_pattern` configration file; the specified name can also be a template that contains tags substituted by, for example, the executable filename, the process ID, or the reason for the dump.[9]
- System-wide dumps on modern Unix-like systems often appear as `vmcore` or `vmcore.incomplete`.
- Systems such as Microsoft Windows, which use filename extensions, may use extension `.dmp`; for example, core dumps may be named `memory.dmp` or `\Minidump\Mini051509-01.dmp`.

### Windows memory dumps

Microsoft Windows supports two memory dump formats, described below.

#### Kernel-mode dumps

There are three types of kernel-mode dumps:

- Complete memory dump – contains full physical memory for the target system.
- Kernel memory dump – contains all the memory in use by the kernel at the time of the crash.
- Small memory dump – contains various info such as the stop code, parameters, list of loaded device drivers, etc.

To analyze the Windows kernel-mode dumps Debugging Tools for Windows are used.[10]

#### User-mode memory dumps

User-mode memory dump, also known as *minidump*,[11] is a memory dump of a single process. It contains selected data records: full or partial (filtered) process memory; list of the threads with their call stacks and state (such as registers or TEB); information about handles to the kernel objects; list of loaded and unloaded libraries. Full list of options available in `MINIDUMP_TYPE` enum.[12]

## Space missions

The NASA Voyager program were probably the first craft to routinely utilize the core dump feature in the Deep Space segment. The core dump feature is a mandatory telemetry feature for the Deep Space segment as it has been proven to minimize system diagnostic costs. The Voyager craft uses routine core dumps to spot memory damage from cosmic ray events.

Space Mission core dump systems are mostly based on existing toolkits for the target CPU or subsystem. However, over the duration of a mission the core dump subsystem may be substantially modified or enhanced for the specific needs of the mission.

## See also

- Database dump

- Hex dump

## References

1. "AIX 7.1 information".
2. `core(4) (https://docs.oracle.com/cd/E26505_01/html/816-5174/core-4.html)` : Process core file – Solaris 10 File Formats Reference Manual
3. Oxford English Dictionary, *s.v.* 'core'
4. Cory Janssen. "What is a Database Dump? - Definition from Techopedia". *Techopedia.com*. Retrieved 29 June 2015.
5. "How to configure a computer to capture a complete memory dump". *sophos.com*. 12 July 2010. Retrieved 29 June 2015.
6. "storage dump definition".
7. Venkateswaran, Sreekrishnan (2008). *Essential Linux device drivers*. Prentice Hall open source software development series. Prentice Hall. p. 623. ISBN 978-0-13-239655-4. Retrieved 2010-07-15. "Until the advent of kdump, *Linux Kernel Crash Dump* (LKCD) was the popular mechanism to obtain and analyze dumps."
8. Fedora Documentation Project (2010). *Fedora 13 Security Guide*. Fultus Corporation. p. 63. ISBN 978-1-59682-214-6. Retrieved 2010-09-29. "Remote memory dump services, like `netdump`, transmit the contents of memory over the network unencrypted."
9. "core(5) – Linux manual page". *man7.org*. 2015-12-05. Retrieved 2016-04-17.
10. "Getting Started with WinDbg (Kernel-Mode)". Retrieved 30 September 2014.
11. "Minidump Files". Retrieved 30 September 2014.
12. "MINIDUMP_TYPE enumeration". Retrieved 30 September 2014.

## External links

Descriptions of the file format:

- `core(5) (http://www.kernel.org/doc/man-pages/online/pages/man5/core.5.html)` – Linux Programmer's Manual – File Formats
- `core(4) (https://docs.oracle.com/cd/E26505_01/html/816-5174/core-4.html)` – Solaris 10 File Formats Reference Manual
- `core(4) (https://web.archive.org/web/20091113011435/http://docs.hp.com/en/B2355-90680/core.4.html)` – HP-UX 11i File Formats Manual
- `core(5) (https://www.freebsd.org/cgi/man.cgi?query=core&sektion=5)` – FreeBSD File Formats Manual
- `core(5) (http://man.openbsd.org/?query=core&sec=5)` – OpenBSD File Formats Manual
- `core(5) (http://netbsd.gw.com/cgi-bin/man-cgi?core+5+NetBSD-current)` – NetBSD File Formats Manual
- `core(5) (https://developer.apple.com/library/mac/documentation/Darwin/Reference/ManPages/man5/core.5.html)` – Darwin and macOS File Formats Manual
- Minidump files (http://msdn2.microsoft.com/en-us/library/ms680378(VS.85).aspx)

Kernel core dumps:

- `savecore(1M) (https://docs.oracle.com/cd/E26505_01/html/816-5166/savecore-1m.html)` – Solaris 10 System Administration Commands Reference Manual
- Apple Technical Note TN2118: Kernel Core Dumps (http://developer.apple.com/library/mac/#technotes/tn2004/tn2118.html)

Retrieved from "https://en.wikipedia.org/w/index.php?title=Core_dump&oldid=758269979"

Categories: Debugging | Computer errors

- This page was last modified on 4 January 2017, at 13:02.
-