gcore(1)
A NEW UTILITY ON HP-UX 11i

DIGANTA KALITA,
HEWLETT-PACKARD,ESDI,
29,CUNNINGHUM ROAD,
BANGALORE-560052,
INDIA

Phone: (91)(080)2251554 ext1295
diganta@india.hp.com
Fax (91)(80)220-0196

AGENDA

- Abstract
- Introduction
- Features
- Usage
- Steps of *gcore* execution
- Implementation Details
- Contents of core file
- How to analyze the core, dumped by gcore ?
- Applications of *gcore*
- Future enhancements on *gcore*
- Bibliography

ABSTRACT

gcore(1) is a new utility to dump the core image of a running process, without killing it.

INTRODUCTION

In the existing HP-UX scenario

Analysis of

- **inter-related processes simultaneously**
- **mission critical applications,**
- **daemon processes,**
- **interactive or terminal dependent programs**

is very difficult.

Analysis of such processes require core dump of the process(es). *gcore(1)* is a new utility to dump the core of single/multiple process(es) at a time, without killing them.

This utility will be available from HP-UX 11.11 (11i) onwards.

FEATURES

- Thread-safe.
- Dumping multiple processes at a time.
- Interface compatibility.
- Support for different object file formats. (SOM/ELF)
- Support for 32/64 bit architecture.
- *gcore* assures almost 100% uptime of applications.

USAGE

gcore [-o filename]process-id
[[-o filename]process-id...]

By default, *gcore* accepts process-id as an argument and dumps the core in the file "core.<process-id>". The core can be dumped in any other file <filename>, using -o option. Multiple process-ids can also be provided on command line, to dump the core images of multiple processes at a time.

Note: Real and effective user ids of the user must be same as that of process to be dumped. Moreover the filename where core has to be dumped should have write permission for the user.

STEPS OF *gcore* EXECUTION

'gcore process-id'

↓

gcore attaches with the process

↓

process stops execution

↓

gcore dumps the core image of the process

↓

gcore detaches from the process

↓

process resumes execution

IMPLEMENTATION DETAILS

'gcore process-id'

↓

ttrace with TT_PROC_ATTACH argument as request and TT_DETACH_ON_EXIT argument as address. If this call succeeds, it will stop the execution of the running process.

↓

ttrace with TT_PROC_CORE argument as request. This call will dump the core.

↓

ttrace with TT_PROC_DETACH argument as request. After this request process resumes execution.

CONTENTS OF CORE FILE

The core file contains all the process information pertinent to debugging: contents of hardware registers, process status and process data. The format of a core file is object format specific.

For an ELF executable, the core file generated is in ELF format, which contains ELF program and file headers. The program header contains an entry for every segment that was part of the process address space, including shared library segments. The contents of the segments themselves are also part of the core image.

CONTENTS OF CORE FILE (Contd.)

For a SOM executable, the core file generated is in SOM format. Core file consist of objects that represent different segments of a process. Each object is preceded by a corehead data structure, which describes the segment type, virtual memory address of the segment in the process, and length of that segment.

SEGMENTS OF CORE FILE FOR ELF AND SOM EXECUTABLES

segments of core file for SOM executable

CORE_DATA

CORE_EXEC

CORE_FORMAT

CORE_KERNEL

CORE_PROC

CORE_STACK

segments of core file for ELF executable

PT_HP_CORE_NONE

PT_HP_CORE_VERSION

PT_HP_CORE_KERNEL

PT_HP_CORE_COMM

PT_HP_CORE_PROC

PT_HP_CORE_LOADABLE

PT_HP_CORE_STACK

PT_HP_CORE_SHM

PT_HP_CORE_MMF

HOW TO ANALYSE THE CORE, DUMPED BY *gcore*?

The core dumped by *gcore* utility is very similar to the core dumped by signals SIGQUIT, SIGSEGV, SIGBUS etc. So any user level debugger or kernel level debugger can be used to analyze the core file.

APPLICATIONS OF *gcore*

- Facilitates debugging of
  - Interactive/terminal dependent programs (vi, ed, ex, top, shell).
  - Daemon processes.
  - Inter-related processes simultaneously.

- Analysing different instances of a hung process.

- Analysing realtime applications e.g. telecom and aircraft applications.

FUTURE ENHANCEMENTS

- Option for dumping corefile in the readable format.

  eg. gcore -r pid

- Option for automatic invocation of the debugger, after dumping core.

  eg. gcore -d <debugger-name>

- Invoking with <command-name> as the argument, to dump all <command-name> related processes

  eg. gcore vi (dump all the vi related processes at that time).

- Thread implementation for dumping core images of multiple processes at a time.