Linux 下如何产生core文件（core dump设置） - 西门吹雪\_\_\_ - 博客园.mhtml

(1条消息)linux-centos 程序崩溃没有core文件？\_gaga392464782的专栏-CSDN博客.mhtml

**(1条消息)core文件设置，gdb的基本调试，gdb反汇编调试\_z296671124的博客-CSDN博客.mhtml**

# 1 设置core文件带core掉进程的PID

[uxdb@localhost corefile]$ **cat /proc/sys/kernel/core\_uses\_pid**

1

如果不是1，root用户执行echo 1 > **/proc/sys/kernel/core\_uses\_pid**

# 2 设置core文件保存路径和文件名格式

[uxdb@localhost corefile]$

[uxdb@localhost corefile]$ **cat /proc/sys/kernel/core\_pattern**

/home/uxdb/uxdbinstall/dbsql/bin/core-%e-%p-%t

echo “**/home/uxdb/uxdbinstall/dbsql/bin/**core-%e-%p-%t“ > /proc/sys/kernel/core\_pattern

%e core掉的程序名称

%p 进程PID

%t core发生的时间

为了永久生效 把 echo “**/home/uxdb/uxdbinstall/dbsql/bin/**core-%e-%p-%t“ > /proc/sys/kernel/core\_pattern

拷贝到/etc/profile最后一行

参考第4章

# 1-2 /etc/sysctl.conf配置 确保重启后设置生效

vim /etc/sysctl.conf

kernel.core\_pattern = **/home/uxdb/uxdbinstall/dbsql/bin/core-%e-%p-%t**

kernel.core\_uses\_pid = **1**

# 3 对于postgres服务进程 必须设置

echo "1" > /proc/sys/fs/suid\_dumpable （出于安全考虑，suid程序以及调用setuid()的程序缺省情况下不产生core dump）

为了永久生效 把echo "1" > /proc/sys/fs/suid\_dumpable 拷贝到/etc/profile最后一行

参考第4章

# 4 core文件尺寸大小限制设置

[uxdb@localhost bin]$ ulimit -a

**core file size (blocks, -c) unlimited**

data seg size (kbytes, -d) unlimited

scheduling priority (-e) 0

file size (blocks, -f) unlimited

pending signals (-i) 7168

max locked memory (kbytes, -l) 64

max memory size (kbytes, -m) unlimited

open files (-n) 1024

pipe size (512 bytes, -p) 8

POSIX message queues (bytes, -q) 819200

real-time priority (-r) 0

stack size (kbytes, -s) 8192

cpu time (seconds, -t) unlimited

max user processes (-u) 4096

virtual memory (kbytes, -v) unlimited

file locks (-x) unlimited

[uxdb@localhost bin]$

如果**core file size 不是 如上面所示的unlimited，则如下设置**

vi /etc/profile

在文件最后 添加 下面2行

ulimit -c unlimited

echo "1" > /proc/sys/fs/suid\_dumpable

保存 退出vi

source /etc/profile

# abrt服务（生成core文件的服务）配置

/etc/abrt/abrt-action-save-package-data.conf

文件内容 加粗字体 必须设置相同

[uxdb@localhost corefile]$ **cat /etc/abrt/abrt-action-save-package-data.conf**

# With this option set to "yes",

# only crashes in signed packages will be analyzed.

# the list of public keys used to check the signature is

# in the file gpg\_keys

#

# How can I check the GPG key used to sign an installed pacakge on

# Red hat Enterprise Linux:

# https://access.redhat.com/solutions/1120013

#

# Product Signing (GPG) Keys:

# https://access.redhat.com/security/team/key

#

OpenGPGCheck = **no**

# Blacklisted packages

#

BlackList = nspluginwrapper, valgrind, strace, mono-core

# Process crashes in executables which do not belong to any package?

#

ProcessUnpackaged = **yes**

# Blacklisted executable paths (shell patterns)

#

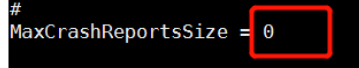
BlackListedPaths = /usr/share/doc/\*, \*/example\*, /usr/bin/nspluginviewer, /usr/lib/xulrunner-\*/plugin-container

# interpreters names

Interpreters = python2, python2.7, python, python3, python3.3, perl, perl5.16.2

[uxdb@localhost corefile]$

/etc/abrt/abrt.conf 文件



# 查看生成的core文件

uxdb为执行程序文件名，执行gdb后，在gdb里有时候需要再执行share

gdb -c core-uxdb-50986-1541387789 uxdb

[uxdb@localhost bin]$ **gdb -c core-uxdb-56850-1578994503 uxdb**

GNU gdb (GDB) 8.1

Copyright (C) 2018 Free Software Foundation, Inc.

License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>

This is free software: you are free to change and redistribute it.

There is NO WARRANTY, to the extent permitted by law. Type "show copying"

and "show warranty" for details.

This GDB was configured as "x86\_64-pc-linux-gnu".

Type "show configuration" for configuration details.

For bug reporting instructions, please see:

<http://www.gnu.org/software/gdb/bugs/>.

Find the GDB manual and other documentation resources online at:

<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".

Type "apropos word" to search for commands related to "word"...

Reading symbols from uxdb...done.

[New LWP 56850]

[Thread debugging using libthread\_db enabled]

Using host libthread\_db library "/lib64/libthread\_db.so.1".

Core was generated by `uxdb: uxdb regression [local] SELECT '.

Program terminated with signal SIGSEGV, Segmentation fault.

#0 0x0000000000a2b988 in **MultiExecBitmapOr** (node=0x2933ce8) at **nodeBitmapOr.c:146**

146 if (RelationIsAoCols(((BitmapIndexScanState \*)subnode)->ss.ss\_currentRelation)) {

(gdb)

(gdb)

(gdb)

(gdb) bt

#0 0x0000000000a2b988 in **MultiExecBitmapOr** (node=0x2933ce8) at nodeBitmapOr.c:146

#1 0x0000000000a1afc3 in MultiExecProcNode (node=0x2933ce8) at execProcnode.c:534

#2 0x0000000000a2be66 in BitmapHeapNext (node=0x28fbf50) at nodeBitmapHeapscan.c:120

#3 0x0000000000a1c5b9 in ExecScanFetch (node=0x28fbf50, accessMtd=0xa2bd69 <BitmapHeapNext>, recheckMtd=0xa2cf6e <BitmapHeapRecheck>) at execScan.c:99

#4 0x0000000000a1c628 in ExecScan (node=0x28fbf50, accessMtd=0xa2bd69 <BitmapHeapNext>, recheckMtd=0xa2cf6e <BitmapHeapRecheck>) at execScan.c:149

#5 0x0000000000a2cfec in ExecBitmapHeapScan (pstate=0x28fbf50) at nodeBitmapHeapscan.c:738

#6 0x0000000000a1aeb1 in ExecProcNodeFirst (node=0x28fbf50) at execProcnode.c:466

#7 0x0000000000a24f85 in ExecProcNode (node=0x28fbf50) at ../../../src/include/executor/executor.h:322

#8 0x0000000000a25366 in fetch\_input\_tuple (aggstate=0x28fb728) at nodeAgg.c:688

#9 0x0000000000a27c93 in agg\_retrieve\_direct (aggstate=0x28fb728) at nodeAgg.c:2313

#10 0x0000000000a278e2 in ExecAgg (pstate=0x28fb728) at nodeAgg.c:2124

#11 0x0000000000a1aeb1 in ExecProcNodeFirst (node=0x28fb728) at execProcnode.c:466

#12 0x0000000000a1367a in ExecProcNode (node=0x28fb728) at ../../../src/include/executor/executor.h:322

#13 0x0000000000a16008 in ExecutePlan (estate=0x28fb518, planstate=0x28fb728, use\_parallel\_mode=0 '\000', operation=CMD\_SELECT, sendTuples=1 '\001', numberTuples=0,

direction=ForwardScanDirection, dest=0x291d8e0, execute\_once=1 '\001') at execMain.c:2031

#14 0x0000000000a13b47 in standard\_ExecutorRun (queryDesc=0x29216c8, direction=ForwardScanDirection, count=0, execute\_once=1 '\001') at execMain.c:372

#15 0x0000000000a139e8 in ExecutorRun (queryDesc=0x29216c8, direction=ForwardScanDirection, count=0, execute\_once=1 '\001') at execMain.c:315

#16 0x0000000000be151b in PortalRunSelect (portal=0x288df18, forward=1 '\001', count=0, dest=0x291d8e0) at uxsqluery.c:932

#17 0x0000000000be11f5 in PortalRun (portal=0x288df18, count=9223372036854775807, isTopLevel=1 '\001', run\_once=1 '\001', dest=0x291d8e0, altdest=0x291d8e0, completionTag=0x7ffe6f888ee0 "")

at uxsqluery.c:773

#18 0x0000000000bdb184 in exec\_simple\_query (query\_string=0x28ee038 "SELECT count(\*) FROM bmscantest WHERE a = 1 OR b = 1;") at uxdb.c:1117

#19 0x0000000000bdf6e9 in UxdbMain (argc=1, argv=0x289aab0, dbname=0x289a898 "regression", username=0x2893ee8 "uxdb") at uxdb.c:4250

#20 0x0000000000b420ef in BackendRun (port=0x2865300) at uxmaster.c:4753

#21 0x0000000000b4186c in BackendStartup (port=0x2865300) at uxmaster.c:4425

#22 0x0000000000b3dc49 in ServerLoop () at uxmaster.c:1990

#23 0x0000000000b3d1cb in PostmasterMain (argc=8, argv=0x2866080) at uxmaster.c:1566

#24 0x0000000000a7ab65 in main (argc=8, argv=0x2866080) at main.c:280

162M ./core-uxdb-56375-1578994499

[uxdb@localhost bin]$ pwd

/home/uxdb/uxdbinstall/dbsql/bin