Simple Summary about gcov in Linux Kernel

Rockins Chen<yunchuan.chen.ext@nsn.com>03/25/08

TOC

1 Purpose and scope

gcov is a coverage testing tool. However, it conventionally can only work under user space, other than kernel space. The object of this short essay is to describe how to test code coverage in Linux kernel by gcov.

2 Methodology

Hubertus Franke, Nigel Hinds, Peter Oberparleiter, and Rajan Ravindran have contributed a gcov-kernel extension[1]. So all we need to do is patch this extension to Linux kernel, re-compile and install the kernel. After that, you can profit from gcov-kernel by testing code coverage in kernel space.

I experimented again and again, and finally found that Linux 2.6.14 is best suitable both for my Hardware platform. If you are extending your kernel with gcov, this version of Linux is suggested.

3 Patch and compile Linux kernel

To extend your Linux kernel, you'd have to get a kernel source tarball[2](version 2.6.14 suggested). Of course, you need to get a copy of gcov-kernel extension tarball from [1]. There's another tool called lcov, which can capture the gcov's coverage data and generate html output. This tool is helpful, you can download it from [1], too.

As to my case, the downloaded tartalls are:

- linux-2.6.14.tar.bz2
- linux-2.6.14-gcov.patch.tgz
- lcov-1.6.tar.gz

They all are putted under /home/rockins/src/ directory.

Now extract source trees from tarballs and patch the kernel. As follows:

```
[root@chn src]# tar jxvf linux-2.6.14.tar.bz2
[root@chn src]# tar zxvf linux-2.6.14-gcov.patch.tgz
[root@chn src]# patch -p0 < linux-2.6.14-gcov.patch
```

Then change current directory to linux-2.6.14 and configure:

```
make menuconfig
```

There's a first-level option:

```
GCOV coverage profiling
```

Enter in and make sure to toggle following two options on(builtin):

```
[*] Include GCOV coverage profiling
[ ]   Profile entire Kernel
<*>   Provide GCOV proc file system entry
[ ]   Support for modified GCC version 3.3.x (hammer patch)
```

Further, you should enable /proc file system support, since gcov-kernel extension output gcov data through the file system.

After configuration, you can compile and install the patched kernel now:

```
[root@chn linux-2.6.14]# make all
[root@chn linux-2.6.14]# make modules_install install
```

If all go through, congratulations, you have gcov extended Linux kernel.

4 Compile kernel module

To test code coverage of a kernel module, some compiling options should be toggled on: -fprofile-arcs and -ftest-coverage. Following two bold lines are added to Makefile of a kernel module manually by me:

```
ifneq ($(KERNELRELEASE),)

obj-m   := rtp.o

GCOV_FLAGS  = -fprofile-arcs -ftest-coverage
EXTRA_CFLAGS += $(GCOV_FLAGS)
...
else
...
endif
```

Very Very Important: Linux 2.6.x kernel build system divides the Makefile into two parts: kbuild and Makefile.

```
ifneq ($(KERNELRELEASE),)
...        /* kbuild part */
else
...        /* Makefile */
endif
```

The gcov flags must located in **kbuild** part, or else it will have not any affect at all to the module.

Then compile the kernel module and copy it to kernel modules directory:

```
[root@chn Forwarder]# make
[root@chn Forwarder]# cp rtp.ko /lib/modules/2.6.14-gcov/kernel/net/
```

After compiling, there's a file named .tmp_rtp.gcno, you should change its name to rtp.gcno(the reason will be stated later):

```
[root@chn Forwarder]# mv .tmp_rtp.gcno rtp.gcno
```

After this step, the gcov-kernel extension is ready.

5 generate .info and generate .html

To generate .html for gcov data, lcov is essential. Extract the source code and install it:

```
[root@chn src]# tar zxvf lcov-1.6
[root@chn src]# cd lcov-1.6
[root@chn lcov-1.6]# make install
```

Theoretically, lcov can capture gcov-kernel data from /prog/gcov/ automatically. However, I found there are some problems with our rtp.ko module. The problem can be described as: In normal situation, for rtp.c, gcc will generate a file rtp.gcno, used by gcov. But, unfortunately, Linux kernel build system add a .tmp_ prefix to this file, result in . tmp_rtp.gcno file generated. The problem will propagate to /proc/gcov/. In my scenario, the files under /proc/gcov/module/root/.../Forwarder/ looks like:

```
[root@chn Forwarder]# pwd
/proc/gcov/module/root/.../Forwarder
[root@chn Forwarder]# ls -a
.  ..  .tmp_rtp.c  .tmp_rtp.gcda  .tmp_rtp.gcno
```

Clearly it's not right. Then what can we do? I found a way to work around this problem:

First, copy the .gcda file under /proc/gcov/ to a different place(Here I use / root/Forwarder-gcov/). For example:

```
[root@chn Forwarder-gcov]# pwd
/root/Forwarder-gcov
[root@chn Forwarder-gcov]# cp /proc/gcov/module/root/.../Forwarder/.tmp_rtp.gcda ./
rtp.gcda
```

And then link the .c and .gcno files under the source tree of the kernel module to the same place of rtp.gcda:

```
[root@chn Forwarder-gcov]# ln -s /root/.../Forwarder/rtp.c .
[root@chn Forwarder-gcov]# ln -s /root/.../Forwarder/rtp.gcno .
```

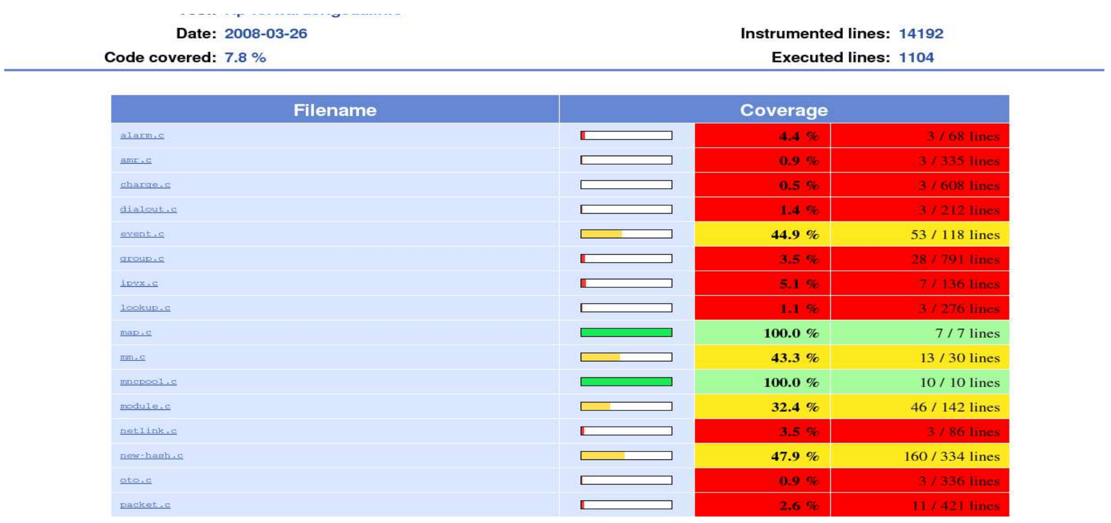
Run following command:

```
[root@chn Forwarder-gcov]# geninfo /root/Forwarder-gcov/
```

This will generate a file named rtp.gcda.info under /root/Forwarder-gcov/. Subsequently, try to generate html:

```
[root@chn Forwarder-gcov]# genhtml rtp.gcda.info
```

This will create an index.html under /root/Forwarder-gcov/, and you can view it in FireFox. Following is a snapshot:



Date:	2008-03-26	Instrumented lines:	14192
Code covered:	7.8 %	Executed lines:	1104

Filename	Coverage		
alarm.c		4.4 %	3 / 68 lines
amr.c		0.9 %	3 / 335 lines
charge.c		0.5 %	3 / 608 lines
dialout.c		1.4 %	3 / 212 lines
event.c		44.9 %	53 / 118 lines
group.c		3.5 %	28 / 791 lines
ipvx.c		5.1 %	7 / 136 lines
lookup.c		1.1 %	3 / 276 lines
map.c		100.0 %	7 / 7 lines
mm.c		43.3 %	13 / 30 lines
mncpool.c		100.0 %	10 / 10 lines
module.c		32.4 %	46 / 142 lines
netlink.c		3.5 %	3 / 86 lines
new-hash.c		47.9 %	160 / 334 lines
oto.c		0.9 %	3 / 336 lines
packet.c		2.6 %	11 / 421 lines

6 Important Note

1. lcov is not used explicitly, but geninfo and genhtml do. Actually, lcov call them implicitly;
2. If geninfo prompts you following warnings:

```
/root/.../Forwarder/rtp.gcda:stamp mismatch with graph file
WARNING: gcov did not create any files for /root/Forwarder-gcov/rtp.gcda!
```

You should kill the application which is using rtp.ko and unload rtp.ko module first, then re-run you application again.

3. Pay a lot of attentions to Makefile of your kernel module(the gcov's option should be added to the kuild part, other than Makefile part). Wrong Makefile won't produce required gcov data.

7 Appendix: lcov for user space

For user space program, lcov will be used. Likewise, you should supply -fprofile-arcs and -ftest-coverage flags to gcc when compiling your program. Here I'll give an example:

```
[root@chn ccplayer-gcov]# lcov -c -b /root/workspace/ccplayer/Debug/ -d /root/workspace/ccplayer/Debug/home/rockins/src/ccplayer/trunk/ -d /root/workspace/ccplayer/Debug/home/rockins/src/ccplayer/trunk/libavi/ -o ccplayer.info
[root@chn ccplayer-gcov]# genhtml ccplayer.info
```

Then, index.html will be generated. Here, -b option is important, since my Makefile is generated under /root/workspace/ccplayer/Debug/ by eclipse automatically. If I do not point out this base directory, following error will emerge:

```
../home/rockins/src/ccplayer/trunk/output.c:cannot open source file
```

A problem related to gcov in user space: It seems that gcov only works if the process exit normally, otherwise no .gcda file generated. Generally, sending SIGTERM to the according process will force it exit normally, and .gcda file will be generated, at that point.

A special tip: If sometimes you end up with such as a linkage error message: '__gcov_init' undefined. You can work around this problem by adding a -lgcov flag to gcc.

8 Conclude and future work

gcov-kernel extension is powerful. But, there's one drawback: coping the .gcda file from /proc/gcov/ and linking source files every time is so annoying, maybe we can write a script to carry this work.

Resource

[1] http://ltp.sourceforge.net/coverage/gcov.php
[2] http://www.kernel.org/pub/linux/kernel/v2.6/

Acknowledge

Thanks to Bill, Simon, Joey gratefully.