1. Oprofile

Oprofile is an excellent tool to analyze the bottleneck of performance of the program. It has two ways to analyze the code. First eventbased can analyze code base on specific event such as Cache miss. Another timedbased analyze the code based on the interrupt of the timer.

1. Oprofile buildup
2. Kernel space

Makesure the oprofile.ko is compiled in the kernel if not rebuild the kernel

CONFIG\_OPROFILE=m

CONFIG\_PROFILING=y

CONFIG\_X86\_LOCAL\_APIC=y

CONFIG\_X86\_IO\_APIC=y

CONFIG\_PCI\_IOAPIC=y

1. User space

Download the Oprofile from <http://oprofile.sourceforge.net/> and install it (./configure make make install)

1. Oprofile tools

1) op\_help：help show.

2) opcontrol：contorl oprofile.

3) opreport：output the result of counting.

4) opannaotate：analyze the source or asm code with counting.

1. Oprofile example

# cat multiply.c

#include <stdio.h>

int fast\_multiply(x, y)

{

return x \* y;

}

int slow\_multiply(x, y)

{

int i, j, z;

for (i = 0, z = 0; i < x; i++)

z = z + y;

return z;

}

int test(x, y)

{

int i, j, z;

for (i = 0, z = 0; i < x; i++)

z = z + y;

return z;

}

int main(int argc, char \*argv[])

{

int i, j;

int x, y, z;

for (i = 0; i < 200; i ++) {

for (j = 0; j < 300; j++) {

x = fast\_multiply(i, j);

y = slow\_multiply(i, j);

z = test(i, j);

}

}

printf("x=%d, y=%d, z=%d\n", x, y, z);

return 0;

}

# gcc multiply.c -g -o multiply

#sudo opcontrol --no-vmlinux //not record kernel part or --vmlinux=/xxx/xx/vmlinux with kernel counting

#sudo modprobe oprofile timer=1 //vmware should be tmier=1, or sudo opcontrol –init on real machine

#sudo opcontrol --start

#./mutiply

#sudo opcontrol --dump

#sudo opcontrol --stop

# opreport -l multiply

Using /var/lib/oprofile/samples/ for samples directory.

CPU: CPU with timer interrupt, speed 2601 MHz (estimated)

Profiling through timer interrupt

samples % symbol name

188 53.4091 test

163 46.3068 slow\_multiply

1 0.2841 fast\_multiply

# opannotate --source ./multiply

Using /var/lib/oprofile/samples/ for session-dir

\* Command line: opannotate --source ./multiply

\*

\* Interpretation of command line:

\* Output annotated source file with samples

\* Output all files

\*

\* CPU: CPU with timer interrupt, speed 2601 MHz (estimated)

\* Profiling through timer interrupt

:

:#include <stdio.h>

:int fast\_multiply(x, y)

:{ // fast\_multiply total: 1 0.2841

: return x \* y;

1 0.2841 :}

:

:int slow\_multiply(x, y)

27 7.6705 :{ // slow\_multiply total: 163 46.3068

: int i, j, z;

135 38.3523 : for (i = 0, z = 0; i < x; i++)

1 0.2841 : z = z + y;

: return z;

:}

:int test(x, y)

40 11.3636 :{ // test total: 188 53.4091

: int i, j, z;

148 42.0455 : for (i = 0, z = 0; i < x; i++)

: z = z + y;

: return z;

:}

:

:int main(int argc, char \*argv[])

:{

: int i, j;

: int x, y, z;

: for (i = 0; i < 200; i ++) {

: for (j = 0; j < 300; j++) {

: x = fast\_multiply(i, j);

: y = slow\_multiply(i, j);

: z = test(i, j);

: }

: }

: printf("x=%d, y=%d, z=%d\n", x, y, z);

: return 0;

:}

#sudo opcontrol --reset// clear the session record.

#sudo opcontrol --shutdown //stop and kill te oprofiled daemon

#sudo opcontrol --deinit//unload the module of oprofile