

How to debug kernel with KGDB on two virtual machines

version 1.0

Pham Tien Nam

© OS TEAM - Updated: 02-12-2011

Content

- ❑ **Overview**
- ❑ Downloads
- ❑ Rebuild kernel with Kgdb patch
- ❑ Setting& Connect 2 virtual machines
- ❑ Debug Kernel use GDB

Overview

- ❑ KGDB is a source level debugger for linux kernel. It is used along with gdb to debug kernel. Current KGDB release is 2.4. Here is list of downloads for current release.

The kgdb patch: [linux-2.6.15.5-kgdb-2.4.tar.bz2](#).
GDB for i386: [gdbmod-2.4.bz2](#)
GDB for x86_64: [x86_64-pc-linux-gdbmod-2.4.tar.bz2](#)

(<http://kgdb.linsyssoft.com/getting.htm>)

- ❑ Two machines (development & target machine) are required for using KGDB. The kernel_debug runs on target machine, GDB runs on development machine. The machines are connected through a serial line. In this case, we use two virtual machines which are created by VMware WorkStation 8

Content

- ❑ Overview
- ❑ **Downloads**
- ❑ Rebuild Kernel with Kgdb patch
- ❑ Setting & connect 2 virtual machines
- ❑ Debug Kernel use GDB

Downloads

❑ List of downloads

Kernel: linux-2.6.15.5.tar.bz2

<http://mirror.anl.gov/pub/linux/kernel/v2.6/>

Kgdb version 2.4 & gdbmod

<http://kgdb.linsyssoft.com/getting.htm>

Vmware WorkStation 8

<http://downloads.vmware.com/d/>

Ubuntu10.10 ISO image to creat Virtual machines

<http://releases.ubuntu.com/10.10/>

Content

- ❑ Overview
- ❑ Downloads
- ❑ **Rebuild Kernel with Kgdb patch**
- ❑ Setting & Connect 2 virtual machines
- ❑ Debug Kernel use GDB

Rebuild Kernel with Kgdb patch(1/3)

Create a virtual machine name "Ubuntu10.10_debug_target"

Copy all downloads(kernel,kgdb,gdb) into folder '/mnt'

Here is commands to rebuild kernel with kgdb patch

```
$ cd /mnt
```

```
$ tar -xvf linux-2.6.15.5.tar.bz2
```

```
$ cd linux-2.6.15.5
```

```
$ tar -xvf ../linux-2.6.15.5-kgdb-2.4.tar.bz2
```

```
$ patch -p1 linux-2.6.15.5-kgdb-2.4/core-lite.patch
```

```
$ patch -p1 linux-2.6.15.5-kgdb-2.4/8250.patch
```

```
$ patch -p1 linux-2.6.15.5-kgdb-2.4/netpoll_pass_skb_to_rx_hook.patch
```

```
$ patch -p1 linux-2.6.15.5-kgdb-2.4/eth.patch
```

```
$ patch -p1 linux-2.6.15.5-kgdb-2.4/i386-lite.patch
```

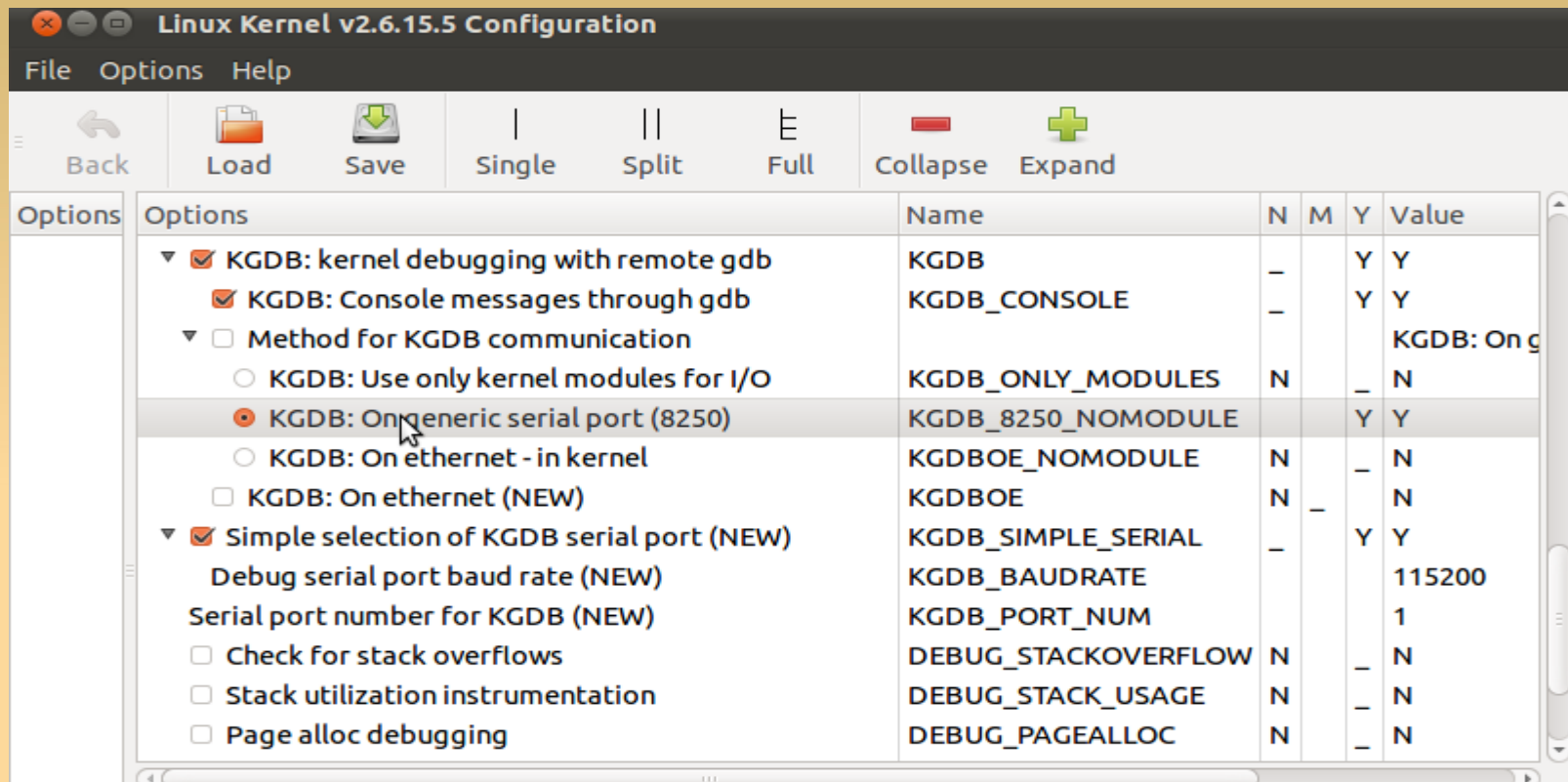
```
$ patch -p1 linux-2.6.15.5-kgdb-2.4/core.patch
```

```
$ patch -p1 linux-2.6.15.5-kgdb-2.4/i386.patch
```

```
$ make mrproper
```

```
$ make gconfig (or: make menuconfig)
```

Rebuild Kernel with Kgdb patch(2/3)



##Save your config then edit Makefile: EXTRAVERSION = -kgdb
\$ make bzImage && make modules && make modules_install && make install

Rebuild Kernel with Kgdb patch(3/3)

Error : "__stack_chk_fail"

```
LD      .tmp.vmlinux1
init/built-in.o: In function `try_name':
/root/Downloads/linux-2.6.15.5/init/do_mounts.c:116: undefined reference to `__stack_chk_fail'
init/built-in.o: In function `name_to_dev_t':
/root/Downloads/linux-2.6.15.5/init/do_mounts.c:207: undefined reference to `__stack_chk_fail'
init/built-in.o: In function `mount_block_root':
/root/Downloads/linux-2.6.15.5/init/do_mounts.c:317: undefined reference to `__stack_chk_fail'
init/built-in.o: In function `change_floppy':
/root/Downloads/linux-2.6.15.5/init/do_mounts.c:359: undefined reference to `__stack_chk_fail'
init/built-in.o: In function `md_setup_drive':
/root/Downloads/linux-2.6.15.5/init/do_mounts/md.c:247: undefined reference to `__stack_chk_fail'
init/built-in.o:/root/Downloads/linux-2.6.15.5/init/initramfs.c:206: more undefined references to `__stack_chk_fail' follow
make: *** [.tmp.vmlinux1] Error 1
```

Solution: Edit Makefile → add '-fno-stack-protector'

```
# Use LINUXINCLUDE when you must reference the include/ directory.
# Needed to be compatible with the O= option
LINUXINCLUDE      := -Iinclude \
                    $(if $(KBUILD_SRC),-Iinclude2 -I$(srctree)/include) \
                    -include include/linux/autoconf.h

CPPFLAGS          := -D__KERNEL__ $(LINUXINCLUDE)

CFLAGS            := -Wall -Wundef -Wstrict-prototypes -Wno-trigraphs \
                    -fno-strict-aliasing -fno-common \
                    -fno-stack-protector \
                    -ffreestanding
AFLAGS            := -D__ASSEMBLY__
```

Content

- ❑ Overview
- ❑ Downloads
- ❑ Rebuild Kernel with KGDB patch
- ❑ **Setting & connect 2 virtual machines**
- ❑ Debug Kernel use GDB

Setting & Connect 2 Virtual Machines(1/3)

Virtual machine 1 “Ubuntu10.10_debug_target”

Setting `/boot/grub/grub.cfg`

```
### BEGIN /etc/grub.d/10_linux ###
menuentry 'Debug kernel 2.6.15.5' --class ubuntu --class gnu-linux --class gnu --class os {
    recordfail
    insmod part_msdos
    insmod ext2
    set root='(hd0,msdos1)'
    search --no-floppy --fs-uuid --set 38178e6f-44e7-486f-b8f4-da24a6480f5f
    linux /boot/vmlinuz-2.6.15-kgdb root=UUID=38178e6f-44e7-486f-b8f4-da24a6480f5f
    ro quiet splash kgdbwait kgdb8250=1,115200
    # initrd /boot/initrd.img-2.6.35-28-generic
}
```

<input checked="" type="checkbox"/> Simple selection of KGDB serial port (NEW)	KGDB_SIMPLE_SERIAL	-	Y	Y
Debug serial port baud rate (NEW)	KGDB_BAUDRATE			115200
Serial port number for KGDB (NEW)	KGDB_PORT_NUM			1

Kgdb8250=1,115200

“115200”: serial port baudrate

“1”: Serial port number for KGDB

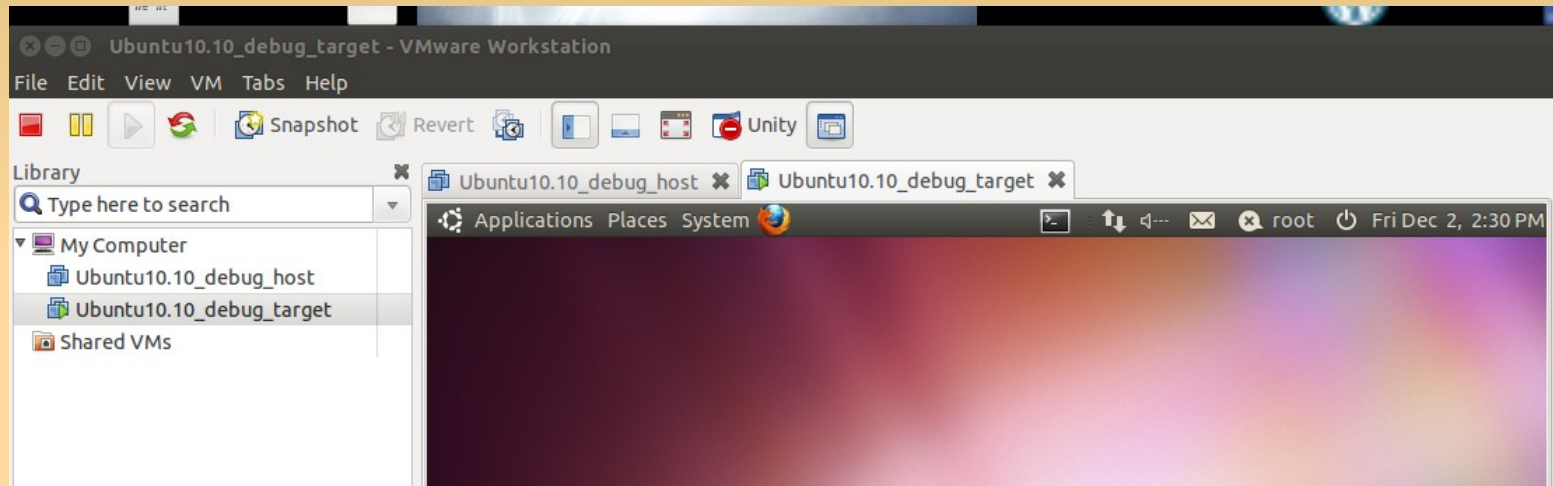
You can choose “0” when you
Rebuild kernel

Setting & Connect 2 Virtual Machines(2/3)

Virtual machine 2 “Ubuntu10.10_debug_host”

Use *vmware-vdiskmanager* to creat VM2 from VM1

```
$vmware-vdiskmanager -r Ubuntu10.10_debug_target.vmdk -t 0 Ubuntu10.10_debug_host.vmdk
```



Add Serial port:

Edit virtual machine settings → Add... → Serial Port -->Output to socket

Setting & Connect 2 Virtual Machines(3/3)

Device Status

☐ Connected

☒ Connect at power on

Connection

☐ Use a physical serial port:

Device:

☐ Use output file:

☒ Use socket (named pipe)

From: To:

I/O Mode

☒ Yield CPU on poll

Allow the guest operating system to use this serial port in polled mode (as opposed to interrupt mode).

Virtual machine 2
Ubuntu10.10_debug_host

Device Status

☐ Connected

☒ Connect at power on

Connection

☐ Use a physical serial port:

Device:

☐ Use output file:

☒ Use socket (named pipe)

From: To:

I/O Mode

☐ Yield CPU on poll

Allow the guest operating system to use this serial port in polled mode (as opposed to interrupt mode).

Virtual machine 1
Ubuntu10.10_debug_target

Content

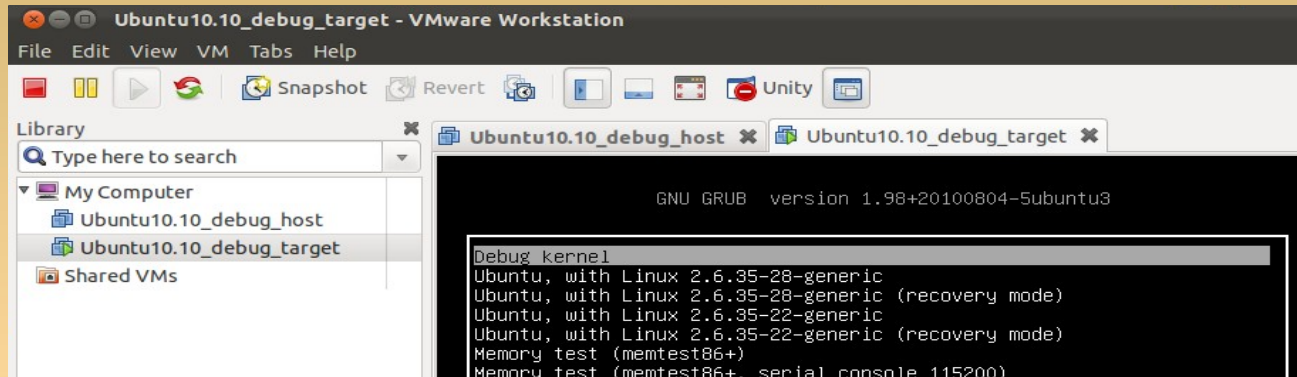
- ❑ Overview
- ❑ Downloads
- ❑ Rebuild Kernel with KGDB patch
- ❑ Setting & connect 2 virtual machines
- ❑ **Debug Kernel using GDB**

Content

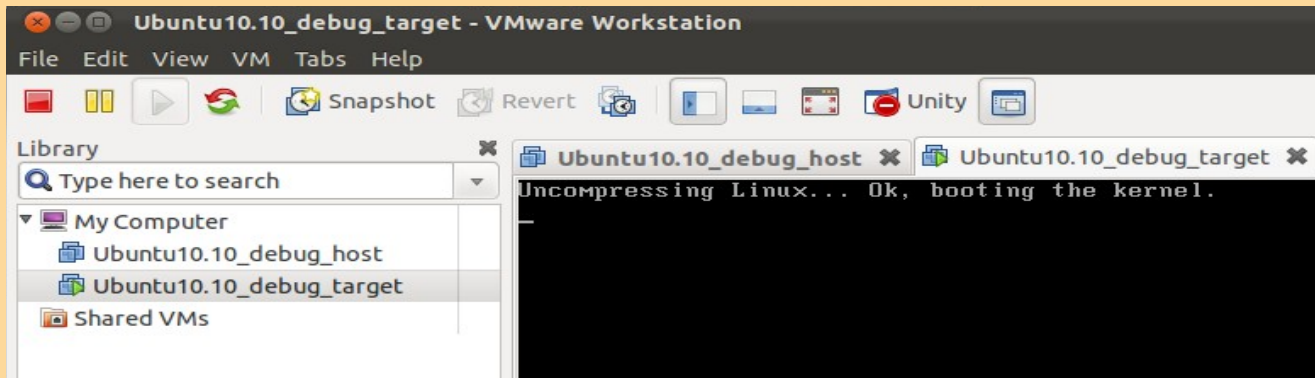
- ❑ Overview
- ❑ Downloads
- ❑ Rebuild Kernel with KGDB patch
- ❑ Setting & connect 2 virtual machines
- ❑ **Debug Kernel using GDB**

Debug Kernel use GDB(1/2)

VM1: Turn on → choose “Debug Kernel”



Waiting Screen



Debug Kernel use GDB(2/2)

VM2: Turn on → choose “Ubuntu.with Linux 2.6.35-28-generic”

Extract “gdbmod-2.4.bz2” to /mnt

Open Terninal

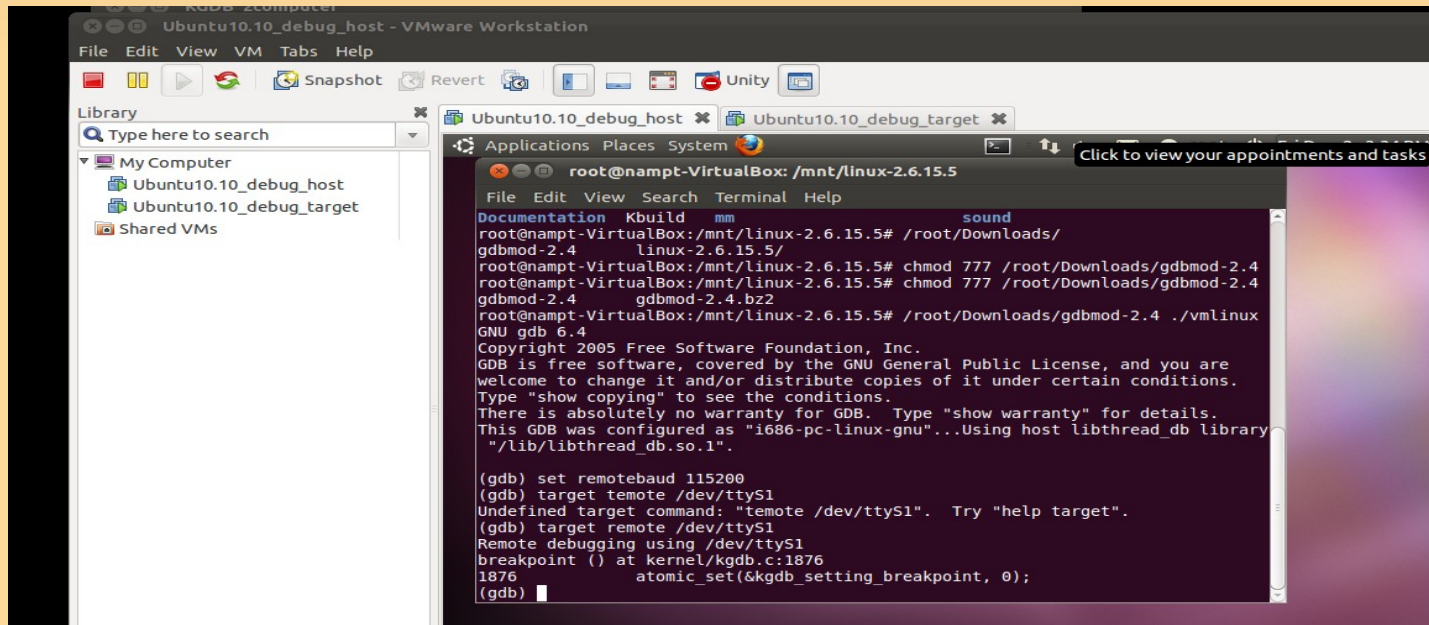
```
$cd /mnt/linux-2.6.15.5
```

```
$/mnt/gdbmod-2.4 ./vmlinux
```

```
$set remotebaud 115200
```

```
$target remote /dev/ttyS1
```

Now creat breakpoint with “break ” then “c” to running



```
root@nampt-VirtualBox: /mnt/linux-2.6.15.5
Documentation Kbuidl mm sound
root@nampt-VirtualBox:/mnt/linux-2.6.15.5# /root/Downloads/
gdbmod-2.4 linux-2.6.15.5/
root@nampt-VirtualBox:/mnt/linux-2.6.15.5# chmod 777 /root/Downloads/gdbmod-2.4
root@nampt-VirtualBox:/mnt/linux-2.6.15.5# chmod 777 /root/Downloads/gdbmod-2.4
gdbmod-2.4 gdbmod-2.4.bz2
root@nampt-VirtualBox:/mnt/linux-2.6.15.5# /root/Downloads/gdbmod-2.4 ./vmlinux
GNU gdb 6.4
Copyright 2005 Free Software Foundation, Inc.
GDB is free software, covered by the GNU General Public License, and you are
welcome to change it and/or distribute copies of it under certain conditions.
Type "show copying" to see the conditions.
There is absolutely no warranty for GDB. Type "show warranty" for details.
This GDB was configured as "i686-pc-linux-gnu"...Using host libthread_db library
"/lib/libthread_db.so.1".

(gdb) set remotebaud 115200
(gdb) target remote /dev/ttyS1
Undefined target command: "remote /dev/ttyS1". Try "help target".
(gdb) target remote /dev/ttyS1
Remote debugging using /dev/ttyS1
breakpoint () at kernel/kgdb.c:1876
1876 atomic_set(&kgdb_setting_breakpoint, 0);
(gdb)
```