

SECTION 10 Configure Kernel Parameters

In this section you learn how to fine-tune kernel parameters.

Objectives

1. [Work with the proc File System](#)
2. [Understand the sys File System](#)
3. [Use the YaST Powertweak Module](#)
4. [Create a Customized initrd](#)

You can use the files in the directory `/proc/` to monitor and modify kernel parameters.

The `sys` file system represents all devices and interfaces of a Linux system.

On SLES 9, there is an alternative to the `sys` file system called Powertweak. This graphical tool allows an easy configuration of tunable kernel parameters. SUSE LINUX offers the Powertweak functionality as a YaST module.

The script `/sbin/mkinitrd` is used to generate a new initial RAM disk.

Objective 1 Work with the proc File System

The files and directories in the directory `/proc/` contain a wealth of information about various aspects of the running system.

Some examples are

- **`/proc/scsi/`**. Information about SCSI devices.
- **`/proc/ide/`**. Information about IDE devices.
- **`/proc/net/`**. Information about network configuration.
- **`/proc/sys/`**. Kernel configuration parameters.

You can modify the files in the directory `/proc/sys/` while the system is operating, affecting the characteristics of the running machine.

The individual files are ASCII text files that can be viewed by using **`cat`** or **`less`**.

For example, the following listing shows information about the first IDE disk:

```
DA3:~ # cat /proc/ide/hda/model
IBM-DJNA-371800
```

To modify individual values, such as to activate routing, use the command **`echo`**; as shown below:

```
DA3:~ # echo 1 > /proc/sys/net/ipv4/ip_forward
```

You can also modify individual values using the command **`sysctl`**; as shown below:

```
DA3:~ # sysctl -w net.ipv4.ip_forward=1
```

Assume you want to deploy an Oracle database. This requires you to check a number of kernel parameters to make sure the requirements for Oracle are met. For example, kernel parameters needed for running Oracle 10g are

- `kernel.shmall = 2097152`
- `kernel.shmmax = 2147483648`
- `kernel.shmmni = 4096`
- `kernel.sem = 250 32000 100 128`
- `fs.file-max = 65536`
- `net.ipv4.ip_local_port_range = 1024 65000`

You can view and set all these parameters in the subdirectories of `/proc/sys/`.

To view all parameters and the current value that can be configured using `sysctl`, enter

```
sysctl -a
```



Before changing any kernel parameters, check whether the parameter is already set to the appropriate value.

You can also set kernel parameters during the boot process. To set kernel parameters during system boot, use the script `/etc/init.d/boot.sysctl`.

The script calls

`sysctl -e -p /etc/sysctl.conf`

to load `sysctl` settings from the file `/etc/sysctl.conf`. This file does not exist by default but will be used as input once it is created.

An example for `/etc/sysctl.conf` is shown below:

```
net.ipv4.ip_forward = 1
net.ipv4.icmp_echo_ignore_broadcasts = 1
fs.file-max = 65535
kernel.shmmax = 2147483648
```

To configure the system to automatically execute the script `/etc/init.d/boot.sysctl` during the boot process, activate it by using the command **insserv**; as shown below:

```
DA3:~ # insserv boot.sysctl
```

Objective 2 Understand the sys File System

In early versions of Linux, it was difficult to determine which interface belonged to which device. This changed with kernel version 2.6 when the `sys` file system was introduced.

`sysfs` is a virtual file system mounted under `/sys/`.

In a virtual file system, there is no physical device that holds the information. Instead, the file system is generated virtually by the kernel.

`sysfs` represents all devices and interfaces of a Linux system.

In `/sys/`, there are four main directories:

- `/sys/bus/` and `/sys/devices/`
- `/sys/class/` and `/sys/block/`

/sys/bus/ and /sys/devices/

These directories contain different representations of system hardware. Devices are represented here.

For example, the following represents a digital camera connected to the USB bus:

```
/sys/bus/usb/devices/1-1/
```

This directory contains several files that provide information about the device.

The following is a listing of the files in this directory:

```
1-1:1.0          bMaxPower          manufacturer
bcdDevice        bNumConfigurations maxchild
bConfigurationValue bNumInterfaces    power
bDeviceClass     detach_state       product
bDeviceProtocol  devnum             serial
bDeviceSubClass  idProduct          speed
bmAttributes     idVendor           version
```

In the example below, you can determine the manufacturer of the device by reading the content from the manufacturer file:

```
DA3:~ # cat manufacturer
OLYMPUS
```

In this case, an Olympus digital camera is connected to the system.

/sys/class/ and /sys/block/

The interfaces of the devices are represented under these two directories.

For example, the interface belonging to the Olympus digital camera is represented by the directory `/sys/block/sda/`.

The directory `sda` is the digital camera accessed like a SCSI hard disk.

The following is the content of the `sda/` directory:

```
DA3:~ # ls /sys/block/sda
dev      queue  removable  size
device  range  sda1       stat
```

The subdirectory `sda1/` represents the interface to the first partition on the camera's memory card.

For example, by reading the content of `/sys/block/sda/sda1/size`, you can determine the size of the partition; as shown below:

```
DA3:~ # cat /sys/block/sda/sda1/size
31959
```

The partition has a size of 31959 512-byte blocks, which is about 16 MB.

To connect an interface with a device, file system links are used. In the Olympus digital camera example, a link exists from the file `/sys/block/sda/device` to the corresponding device; as shown below:

```
DA3:~ # ll /sys/block/sda/device
lrwxrwxrwx 1 root root 0 Aug 17 14:03 device ->
../../../../devices/pci0000:00/0000:00:1d.0/usb1/1-1/1-1:1.0/host0/0:0:0:0
```

In this way, all interfaces of the system are linked with their corresponding devices.

There are also device files in the `/dev/` directory.

These files are needed for applications to access the interfaces of a device.

The name "device file" is a bit misleading, as in our terminology the name "interface file" would be more suitable.

Objective 3 Use the YaST Powertweak Module

SLES 9 offers a special tool for setting kernel parameters: Powertweak.

Powertweak is comprised of the daemon `powertweakd` and a graphical YaST frontend. Powertweak lets you configure kernel parameters conveniently and transparently.

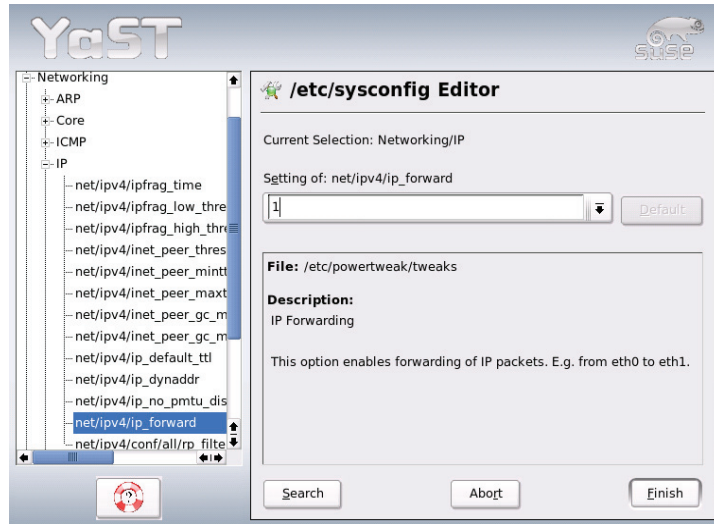
A significant advantage of using Powertweak is that a short description is provided for every parameter.

When Powertweak is started for the first time, the configuration file `/etc/powertweak/tweaks` is generated and the daemon is started.

After this, the daemon is started every time the system is booted, because the links to the start script `/etc/init.d/powertweakd` are also created in the respective runlevel directories in `/etc/init.d/`.

You can also activate routing, as shown below:

Figure 10-1



The result is an entry in the file `/etc/powertweak/tweaks`; as shown below:

```
## Path: Networking/IP
# IP Forwarding
#
# This option enables forwarding of IP packets. E.g. from eth0 to eth1.
#
net/ipv4/ip_forward = 1
```



A disadvantage of having different ways to set kernel parameters is that the same parameter can be set differently in various places.

For instance, changing the variable `net.ipv4.ip_forward = 1` in the file `/etc/sysctl.conf` will not have any effect, if `powertweakd` is started and you set a different value for IP forwarding in the `Powertweak` configuration.

Therefore, you should select one configuration method and use it exclusively to avoid inconsistencies.

Exercise 10-1 **Change Kernel Parameters with Powertweak**

To change kernel parameters with `Powertweak`, complete the following:

1. Ensure that you are logged in to your server's GUI as **geeko** with a password of **N0v3ll**.
2. Launch YaST from the main menu by selecting **System > Configuration > YaST Control Center**.
3. Enter **novell** as root password in the authentication window.
4. On the left, select **System**.
5. On the right, select **Powertweak Configuration**.
6. Search for the parameter **file-max**.
7. Set the value to **65536**.
8. To activate the changes and exit the module, select **Finish**.
9. To leave YaST, select **Close**.
10. Launch a terminal window:
 - a. Press **Alt + F2**.
 - b. Enter **konsole**.

- c. Select **Run**.
11. To get root privileges, enter **sux -** and a password of **novell**.
12. To verify the new parameter value set by Powertweak, enter **cat /proc/sys/fs/file-max**
13. To exit the terminal window, enter **exit** twice.

(End of Exercise)

Objective 4 Create a Customized initrd

Most Linux distributions load an initial RAM disk to the RAM when the system is booted. This RAM disk usually contains kernel modules needed for booting the system.

Like the kernel, the initial RAM disk is located in the `/boot` directory.

This RAM disk is a file system compressed with **gzip**.

It can be decompressed with a command such as **gunzip**.

In the following example, the file `/boot/initrd` is decompressed and stored in the file `/tmp/ramdisk`:

```
DA3:~ # gunzip </boot/initrd >/tmp/ramdisk
```

This uncompressed RAM disk can be mounted on `/mnt/` for testing purposes. The option **-o loop** is required for the mount command; as shown below:

```
DA3:~ # mount -o loop /tmp/ramdisk /mnt
```

Normally, the initial RAM disk does not have to be processed in this way, because the following script automatically performs the required adaptations:

```
/sbin/mkinitrd
```

The corresponding configuration file is `/etc/sysconfig/kernel`. The kernel modules that must be included are specified in the variable `INITRD_MODULES`; as shown below:

```
DA3:~ # head /etc/sysconfig/kernel
## Path:          System/Kernel
## Description:  Modules to load from initrd
## Type:         string
## Command:     /sbin/mkinitrd
#
# This variable contains the list of modules to be added to the initial
# ramdisk by calling the script "mk_initrd"
# (like drivers for scsi-controllers, for lvm or reiserfs)
#
INITRD_MODULES="aic7xxx reiserfs"
```

If you modify the variable `INITRD_MODULES`, you have to execute the command **mkinitrd** to create a new initial RAM disk; as shown below:

```
DA3:~ # mkinitrd
Root device: /dev/hda5 (mounted on / as reiserfs)
Module list: aic7xxx reiserfs
Kernel image: /boot/vmlinuz-2.6.5-7.21-default
Initrd image: /boot/initrd-2.6.5-7.21-default
Shared libs: lib/ld-2.3.3.so lib/libc.so.6
Modules: kernel/drivers/scsi/scsi_mod.ko kernel/drivers/scsi/sd_mod.ko
kernel/drivers/scsi/aic7xxx/aic7xxx.ko /fs/reiserfs/reiserfs.ko
Bootsplash: SuSE-SLES (800x600)
```

Exercise 10-2 Use mkinitrd to Modify the Initial RAM Disk

To use mkinitrd to modify the initial RAM disk, complete the following:

1. Ensure that you are logged in to your server's GUI as **geeko** with a password of **N0v3ll**.
2. Launch a terminal window. Press **Alt + F2**, enter **konsole**, select **Run**.
3. To get root privileges, enter **sux -** and a password of **novell**.
4. To edit the configuration file, enter
vi /etc/sysconfig/kernel
5. Search for the variable **INITRD_MODULES** and add **ext3** as value. As separator, use a blank.
6. To save and exit vi, enter **:wq**.
7. To create a new initial RAM disk, enter **mkinitrd**.
8. Verify that the output includes a line like this:
Module list: reiserfs ext3
9. To open the configuration file again, enter
vi /etc/sysconfig/kernel
10. Search for the value **ext3** you edited before and remove it.
11. To save and exit vi, enter **:wq**.
12. Enter **mkinitrd** again.
13. Verify that the value **ext3** is removed from your initial RAM disk.

(End of Exercise)

Summary

Objective	Summary
1. Work with the proc File System	<p>The <code>/proc/</code> directory contains subdirectories with information about various aspects of the running system:</p> <ul style="list-style-type: none">■ /proc/scsi/. Information about SCSI devices■ /proc/ide/. Information about IDE devices■ /proc/net/. Information about network configuration■ /proc/sys/. Information about kernel configuration parameters <p>The files are ASCII files and can be viewed by using cat or less.</p> <p>To edit values in the files, use echo or sysctl.</p>
2. Understand the sys File System	<p>The <code>sys</code> file system</p> <ul style="list-style-type: none">■ Is a virtual file system mounted under <code>/sys/</code>.■ Represents all devices and interfaces of a Linux system. <p>In <code>/sys/</code>, there are four main directories:</p> <ul style="list-style-type: none">■ /sys/bus/ and /sys/devices/■ /sys/class/ and /sys/block/

Objective	Summary
3. Use the YaST Powertweak Module	You can set kernel parameters by using Powertweak. The configuration file is /etc/powertweak/tweaks. The start script is /etc/init.d/powertweakd.
4. Create a Customized initrd	To create a customized initrd, you need to know the following: The initial RAM disk <ul style="list-style-type: none">■ Is located in the directory /boot/.■ Contains kernel modules needed for system booting. The corresponding configuration file is /etc/sysconfig/kernel. The variable INITRD_MODULES specifies the kernel modules which have to be included. To create a new initial RAM disk, use the command mkinitrd .