1.1 Set up of encrypted root, home and swap partitions

This information was taken from:
http://en.opensuse.org/Encrypted_Root_File_System consulted on 11-05-2008, http://lists.opensuse.org/opensuse/2008-04/msg02199.html consulted on 11-05-2008, and
http://en.opensuse.org/Talk:Encrypted_Root_File_System_with_SUSE_HOWTO#Create_an_encrypted_swap_partition consulted on 11-05-2008.

In order to protect all sensitive data against unauthorized access after theft, the root, swap and home partitions will be encrypted. Encrypting the root partition will provide protection of generally used directories like `/tmp` and `/var`. Encryption of the swap area will also provide protection against attacks on a suspended system.

The root partition will be unlocked using a password. On this partition, the keys for unlocking the swap and home partitions will reside. When booting the system, this setup will prompt the user only once for a password to unlock the root partition and after that, the swap and home partitions will be unlocked automatically.

The harddisk is to be partioned as follows:

75 MB as ext3	/boot	
[amount of physical memory]	/swap	This will become the encrypted swap.
20 GB as ext3	/home	This will become the encrypted root.
remainder as ext3	/	This will become the encrypted home.

In openSUSE 10.3, mkinitrd works out of the box, but sometimes a typo in the script `/lib/mkinitrd/scripts/setup-luks.sh` will have to be corrected.
In the line,

```
    luks_blockdev="$luks_blockdev $lucksbd"
```

replace `$lucksbd` by `$luksbd` (notice the letter "c").

Load the following modules.

```
    modprobe dm-mod
    modprobe dm-crypt
    modprobe aes
    modprobe sha256
    modprobe sha1
```

1.1.1 Encryption of swap partition

Turn off the existing swap:

```
    swapoff [path to swap partition,e.g. /dev/sda6]
```

Fill the swap partition with random data. This will take about 15 minutes.

```
    dd if=/dev/urandom of=[path to swap partition,e.g. /dev/sda6]
```

Create a key for unlocking the swap partition:

```
    touch /etc/crypt-swap.key
    chmod go-rwx /etc/crypt-swap.key
    head -c 2048 /dev/urandom > /etc/crypt-swap.key
```

Initialize the encrypted swap device:

```
    cryptsetup -v --key-size 256 luksFormat [path to swap
    partition,e.g. /dev/sda6] /etc/crypt-swap.key
```

Create a file named `/etc/crypttab` and put these lines in it:

```
    # target device   mount point   key file              options
    cr_sda6           /dev/sda6     /etc/crypt-swap.key   swap
```

`cr_sda6` is just an arbitrary name and `/dev/sda6` should be replaced by the appropriate path to the swap partition.

Edit /etc/fstab and replace the line for the swap partition by:

```
    /dev/mapper/cr_sda6   swap   swap   defaults   0 0
```

Turn on the boot.crypto script. This will automatically format the `/dev/mapper/cr_sda6` device with the swap filesystem based on the `/etc/crypttab` file configuration.

```
    chkconfig boot.crypto on
```

Then test it with:

```
    /etc/init.d/boot.crypto start
    swapon -a
    free
```

The last command should give the amount of swap memory present on the system.

Then reboot the system in order to verify that the swap partition is opened without providing a password. If this fails, start all over again.

1.1.2 Encryption of root partition

Unmount the home partition:

```
    umount [path to the home partition,e.g. /dev/sda7]
```

Fill [path to the home partition,e.g. /dev/sda7] with random data. Depending on the size of the partition and the speed of the CPU, this will take a couple of hours.)

```
    dd if=/dev/urandom of=[path to the home partition,e.g. /dev/sda7]
```

Once the dd command has finished, create the device mapping for the root partition following the same procedure for swap: (be sure to use the same password to enable a single sign-on)

```
    cryptsetup -v --key-size 256 luksFormat [path to the home
    partition,e.g. /dev/sda7]
```

The creation of the encrypted partition must be confirmed and a password or passphrase has to be provided twice.

Now open the encrypted partition:

```
    cryptsetup luksOpen [path to the home partition,e.g. /dev/sda7] root
```

and provide the password or passphrase entered previously.

In order to create a file system on the new device, issue the command:

```
    /sbin/mkfs.ext3 -O dir_index,resize_inode /dev/mapper/root
```

Once, the new encrypted system is created, the complete root file system must be copied to this new file system. First it has to be mounted somewhere:

```
    mkdir /mnt/root
    mount /dev/mapper/root /mnt/root
```

Copy all files from the current root to the encrypted root while preserving permissions, symbolic links and other special files.

```
    cd /
    find bin dev etc home lib* opt root sbin srv \
        tmp usr var -depth -print0 | cpio -pmd \
        --null /mnt/root
```

Create a number of directories that will be used as mount points.

```
    mkdir /mnt/root/proc
    mkdir /mnt/root/sys
    mkdir /mnt/root/media
    mkdir /mnt/root/mnt
```

Open `/mnt/root/etc/fstab` in an editor. **Be careful not to edit `/etc/fstab` instead because this is not the fstab file that will be used at the next boot!**
Change the entry for [path to the FORMER home partition,e.g. /dev/sda7] into:

```
    /dev/mapper/root      /       ext3    acl,user_xattr    1 1
```

In addition, the entry for [path to the NEW home partition,e.g. /dev/sda8], the former root-partition, should be written as:

```
    /dev/sda8            /home    ext3    acl,user_xattr    1 2
```

In order to use the new encrypted root file system, it must be decrypted when the system is booted. This job is best handled by a specially crafted initial ram disk (initrd). The approach taken here is to modify the driving script, mkinitrd, to automatically create the necessary initrd required for an encrypted root file system.

```
    mkinitrd -d /dev/mapper/root
```

Ignore the error message

```
    Resume device: /dev/sda6
    Device does not exist
    Command failed.
```

This will be corrected by editing the boot menu.

The final step is to create a new entry in the Grub menu for the encrypted partition. Edit the boot menu in the file `/boot/grub/menu.lst` and create a NEW entry which contains the new parameters. It should look like this:

```
    ###Encrypted root###
    title openSUSE 10.3 - encrypted
        root (hd0,0)
        kernel /vmlinuz-VER-default root=/dev/mapper/root
        luks_root=/dev/sda7 luks="root" vga=0x317
        resume=/dev/mapper/cr-swap splash=silent showopts
        initrd /initrd-VER-default
```

The portions in bold italics of this new entry should be taken from the old entry for booting Linux. The device names will have to comply with the partitions created previously.

Close all open files and try to boot using the encrypted partition. You will have to type in your password only once for the root partition. The prompt for this is displayed but after that, several other messages appear which makes it look like as if there is no prompt for entering the password.

1.1.3 Encryption of home partition

Proceed to erase the original root partition and replace it by another encrypted file system:

```
    umount [path to the FORMER root partition,e.g. /dev/sda8]
```

Go `Yast` > `System` > `Partitioner`. Create an encrypted `/home` partition.
It happens that an error message is displayed stating that /etc/fstab could not be updated. If this happens, delete the partition and recreate it without encryption. Then, edit the partition and reformat and encrypt it.

Store the password in a text file `/etc/crypt-home.key`. Be sure not to enter any additional spaces or returns. Check with the `joe` editor.

Enter the following line in /etc/crypttab.

```
    cr_sda8    /dev/sda8    /etc/crypt-home.key    none
```

`cr_sda8` should be replaced by the corresponding entry in `/etc/fstab`

Reboot to test the entire setup.