By Andreas Hilboll
Published: 2009-04-30 19:35

Creating A Fully Encrypted Para-Virtualised Xen Guest System Using Debian Lenny

This document explains how to set up a fully encryptedpara-virtualized XEN instance. In this howto, the host system isrunning Debian Etch, while the guest system to be installed will beusing Debian Lenny.

Introduction

If you are concerned about your privacy, you might want toconsider using hard disk encryption to protect your valuableprivate data from spying eyes. Usually, the easiest way would be touse your distribution's installer to set up a fully encryptedsystem; I think most recent Linux distributions support this. However, when you are using XEN to provide virtualization, thereare situations where you might not want to encrypt your wholecomputer with all guest instances, but instead only encrypt one OSinstance. This howto will deal with exactly this situation. Itassumes that the XEN host system is already up and running.

Preparing the XEN instance

Firstly, we need to create the XEN configuration for the newguest instance. This can easily be done with the script*`xen-create-image`* from the package *`xen-tools`*:

```
xen:~# aptitude install xen-tools
```

Now, we need to 'teach' *`xen-tools`*the existence of Lenny (since, remember, we're using Etch as thehost system):

```
xen:~# ln -s /usr/lib/xen-tools/debian.d
/usr/lib/xen-tools/lenny.d
```

Now, we can create the XEN instance:

```
xen:~# xen-create-image --memory 150M --size 1G
--noswap --ip 10.0.0.1 --hostname crypto.example.com --dist
lenny
```

This last step installs a very basic Debian Lenny guest system.We will use this system to configure encrypted filesystems andeventually copy its contents over to these encryptedfilesystems.

The encrypted filesystem(s) of the new system will all be storedusing LVM. So basically, this is kind of a 'LVM inside LVM': Weneed to create a logical volume on the host system which will bemade available to the guest system as `/dev/sdX`, and inside the encrypted guest system, wewill install LVM using this `/dev/sdX` asphysical volume to store our volume group:

```
xen:~# lvcreate -L24G -n
crypto.example.com_crypt vg0
```

Here we assume that the volume group on the XEN server, whichholds the logical volumes of all the XEN instances is called`vg0`.

By default, `xen-create-image` createsa configuration file `/etc/xen/crypto_unencrypted.cfg`. We need to modifythis to include the additional logical volume, so that it reads asfollows:

```
kernel  = '/boot/vmlinuz-2.6.18-6-xen-amd64'
ramdisk = '/boot/initrd.img-2.6.18-6-xen-amd64'
memory  = '150'
root    = '/dev/sda1'
disk    = [ 'phy:vg0/crypto.example.com_disk,sda1,w', 'phy:vg0/crypto.example.com_crypt,sda2,w' ]
name    = 'crypto.example.com'
vif  = [ 'ip=10.0.0.1' ]
on_poweroff = 'destroy'
on_reboot   = 'restart'
on_crash    = 'restart'
```

So now we're ready to first start into the newly createdsystem:

```
xen:~# xm create -c
/etc/xen/crypto_unencrypted.cfg
```

Preparatory steps inside the temporary XEN guest

After logging in, we need to install necessary components:

```
crypto:~# aptitude install lvm2 cryptsetup
```

Next, we fill the target partition with random data:

```
crypto:~# dd if=/dev/urandom of=/dev/sda2
```

Create the cryptodisk:

```
crypto:~# cryptsetup -c aes-cbc-essiv:sha256 -y
-s256 luksFormat /dev/sda2
```

Enable LVM to handle cryptsetup devices. For this, add thefollowing to the `devices` section of`/etc/lvm/lvm.conf`:

```
types = [ "device-mapper", 16 ]
```

Open the crypto device:

```
crypto:~# cryptsetup luksOpen /dev/sda2
crypt
```

Create the physical volume and the volume group for LVM:

```
crypto:~# pvcreate /dev/mapper/crypt

crypto:~# vgcreate vg-crypt /dev/mapper/crypt
```

Create logical volumes:

```
crypto:~# lvcreate -L1G -nroot vg-crypt

crypto:~# lvcreate -L2G -ntmp vg-crypt

crypto:~# lvcreate -L12G -nvar vg-crypt

crypto:~# lvcreate -L1G -nswap vg-crypt

crypto:~# lvcreate -L3G -nusr vg-crypt
```

Now, when creating the logical volumes, there is not exactly 1Gof space left on the device but slightly more. We use `vgdisplay` to find out exactly how much space isleft and then create the last volume:

```
crypto:~# lvcreate -l255 -nusrlocal vg-crypt
```

And create filesystems:

```
crypto:~# mkfs.ext3 /dev/vg-crypt/root

crypto:~# mkfs.ext3 /dev/vg-crypt/tmp

crypto:~# mkfs.ext3 /dev/vg-crypt/usr

crypto:~# mkfs.ext3 /dev/vg-crypt/usrlocal

crypto:~# mkfs.ext3 /dev/vg-crypt/var

crypto:~# mkswap /dev/vg-crypt/swap
```

You might ask yourself why I am creating so many filesystems?Well, this is supposed to become a pretty secure system (after all,otherwise, all the encryption does not help if my system ishacked), so later I will be following the advice of the*Securing Debian Handbook* (**http://www.debian.org/doc/manuals/securing-debian-howto/**),which however is not covered in this howto.

Mount the newly created filesystems:

```
crypto:~# mkdir /mnt/target

crypto:~# mount /dev/vg-crypt/root /mnt/target/



crypto:~# mkdir /mnt/target/usr/local -p

crypto:~# mkdir /mnt/target/var

crypto:~# mkdir /mnt/target/tmp

crypto:~# mount /dev/vg-crypt/usr /mnt/target/usr
```

```
crypto:~# mount /dev/vg-crypt/usrlocal /mnt/target/usr/local

crypto:~# mount /dev/vg-crypt/var /mnt/target/var

crypto:~# mount /dev/vg-crypt/tmp /mnt/target/tmp
```

And copy the currently running filesystem to the encryptedones:

```
crypto:~# init s

crypto:~# cp -apx / /target/
```

Since we copied the data from a running system, we need to cleanup a bit:

```
crypto:~# /bin/rm -fr /target/tmp/*

crypto:~# /bin/rm -fr /target/proc/*

crypto:~# /bin/rm -fr /target/sys/*

crypto:~# /bin/rm /target/etc/mtab
```

Create the `/target/etc/fstab`:

```
proc                /proc     proc    rw,nodev,nosuid,noexec 0     0
/dev/vg-crypt/root  /         ext3    errors=remount-ro      0     1
/dev/vg-crypt/usr   /usr      ext3    errors=remount-ro      0     1
/dev/vg-crypt/usrlocal /usr/local ext3    errors=remount-ro      0     1
/dev/vg-crypt/var   /var      ext3    errors=remount-ro      0     1
/dev/vg-crypt/tmp   /tmp      ext3    errors=remount-ro      0     1
```

```
/dev/vg-crypt/swap    none      swap    sw              0    0
```

As stated above, here you might want to consider the*Securing Debian Handbook* and apply someadditional security tweaks.

For now, stop the guest system:

```
crypto:~# halt
```

Back in the XEN Host-System ...

We need to install some necessary tools:

```
xen:~# aptitude install cryptsetup
initramfs-tools
```

Now we create a XEN configuration file */etc/xen/crypto_encrypted.cfg* for the encryptedsystem:

```
kernel  = '/boot/vmlinuz-2.6.18-6-xen-amd64'
ramdisk = '/boot/initrd.img-2.6.18-6-xen-amd64_crypt'
memory  = '150'
root    = '/dev/mapper/vg--crypt-root'
disk    = [ 'phy:vg0/crypto.example.com_crypt,sda1,w' ]
name    = 'crypto.example.com'
vif  = [ 'ip=10.0.0.1' ]
on_poweroff = 'destroy'
on_reboot   = 'restart'
on_crash    = 'restart'
```

Now comes the really tricky part about this. We need to create anew initrd image so that the encrypted system actually asks for thedisks Key.

First, we create a file */etc/initramfs-tools/conf.d/cryptroot*:

```
CRYPTROOT=target=crypt,source=/dev/sda1,key=none,lvm=vg--crypt-root
```

Note that even though the volume group which holds the encryptedfilesystems is called *vg-crypt*, we needto 'escape' the '-' with a second dash.

Then, we add the following lines to the file */etc/initramfs-tools/modules*:

```
aes-x86_64
dm-crypt
dm-mod
sha256
```

Next, we backup our existing initrd image, create a new one anddo some renaming:

```
xen:~# mv /boot/initrd.img-2.6.18-6-xen-amd64
/boot/initrd.img-2.6.18-6-xen-amd64_orig

xen:~# update-initramfs -k 2.6.18-6-xen-amd64 -v -c

xen:~# mv /boot/initrd.img-2.6.18-6-xen-amd64
/boot/initrd.img-2.6.18-6-xen-amd64_crypt

xen:~# mv /boot/initrd.img-2.6.18-6-xen-amd64_orig
/boot/initrd.img-2.6.18-6-xen-amd64
```

So now we are ready to start the encrypted XEN guest:

```
xen:~# xm create -c
/etc/xen/crypto_encrypted.cfg
```

If all went well, you will be prompted for the Key to theencrypted volume, and the whole systems boots. Enjoy :)

Now you can still clean up a bit:

```
xen:~# /bin/rm /etc/xen/crypto_unencrypted.cfg

xen:~# lvremove /dev/vg0/crypto.example.com_disk
```

One important thing: YOU NEED TO BE ABLE TOTRUST THE HOST SYSTEM!!! If the host system is compromised, itmight log the console input when you enter the Key for unlockingthe cryptsetup device, thus learning the password whith which youencrypted all your data!!!

Sources

While setting up this encrypted XEN guest instance, thefollowing websites proved useful to me:

- **http://www.debian.org/doc/manuals/securing-debian-howto/**
- **http://tuxmobil.de/samsung_x20_linux_lvm_encryption.html/**
- **http://www.saout.de/tikiwiki/tiki-index.php?page=EncryptedDevice**
- **http://madduck.net/docs/cryptdisk/**