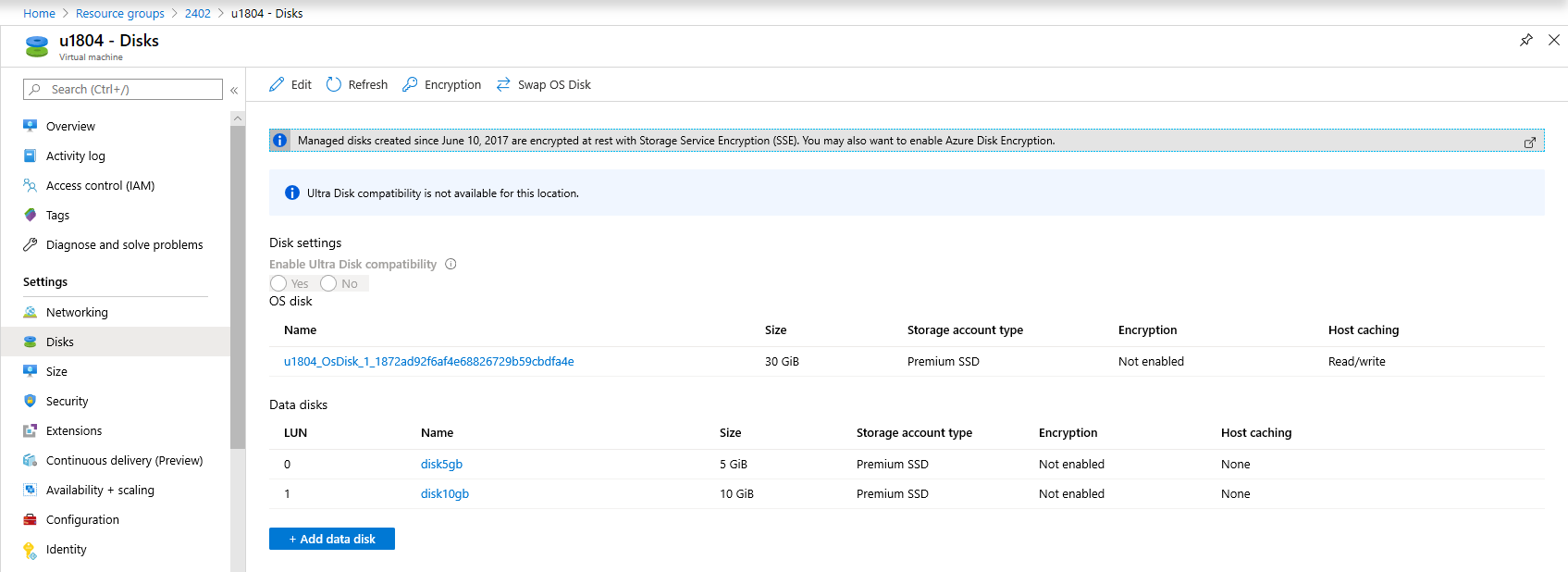
LVM on Crypt

# Preparing the virtual machine

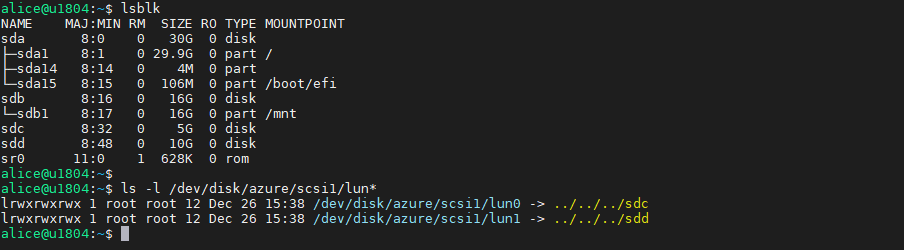
1. Deploy a Ubuntu 18.04 VM in size, Standard D2s v3 (2 vcpu, 8 GB memory), and with 2 data drives (with 5GB and 10GB respectively)



1. In the VM, identify the disks

Lsblk -f

ls -l /dev/disk/azure/scsi1/lun\*



1. Format and mount each Azure LUN that will be used by LVM data VGs and add them to /etc/fstab
   1. Format each of the LUNs that will be used by LVM after the encryption, make sure you use the Azure paths in the form /dev/disk/azure/scsi1/lun# as shown below.

echo "y" | mkfs.ext4 /dev/disk/azure/scsi1/lun0

echo "y" | mkfs.ext4 /dev/disk/azure/scsi1/lun1

* 1. Add them to /etc/fstab using the following commands:

UUID0="$(blkid -s UUID -o value /dev/disk/azure/scsi1/lun0)"

UUID1="$(blkid -s UUID -o value /dev/disk/azure/scsi1/lun1)"

mkdir /data0

mkdir /data1

echo "UUID=$UUID0 /data0 ext4 defaults,nofail 0 0" >>/etc/fstab

echo "UUID=$UUID1 /data1 ext4 defaults,nofail 0 0" >>/etc/fstab

Important: Replace /data0 and /data1 with your corresponding mount points

* 1. Ensure the LUNs are mounted

mount -a

df -h /data\*

# Encrypting the underlying devices

1. Proceed to encrypt the data disks on this VM executing the following PowerShell script:
   1. Enable disk encryption (In this case we are using Powershell, you can also use CLI)

# Modify all the following parameters accordingly

$rgName = 'VM Resource Group Name'

$vmName = 'VM Name'

$rgKeyName = 'Key Vault Resource Group Name'

$KeyVaultName = 'Key Vault Name'

$ADEKeyName=’ADE Key Name’

$VolumeType = 'Data'

# The following line it's only required if KEK is used

$KeyVault = Get-AzKeyVault -VaultName $KeyVaultName -ResourceGroupName $rgKeyName;

$diskEncryptionKeyVaultUrl = $KeyVault.VaultUri

$KeyVaultResourceId = $KeyVault.ResourceId

# The following 2 lines are only required if KEK is used

$key = Get-AzureKeyVaultKey -VaultName $KeyVaultName -Name $ADEKeyName

$keyencryptionkeyurl=$key.Id

$sequenceVersion = [Guid]::NewGuid()

# KEK with EncryptFormatAll

Set-AzVMDiskEncryptionExtension -ResourceGroupName $rgName -VMName $vmName -DiskEncryptionKeyVaultUrl $diskEncryptionKeyVaultUrl -DiskEncryptionKeyVaultId $KeyVaultResourceId -KeyEncryptionKeyUrl $keyencryptionkeyurl -KeyEncryptionKeyVaultId $KeyVaultResourceId -VolumeType $VolumeType -SequenceVersion $sequenceVersion -skipVmBackup -EncryptFormatAll

# Without KEK with EncryptFormatAll

#Set-AzVMDiskEncryptionExtension -ResourceGroupName $rgName -VMName $vmName -DiskEncryptionKeyVaultUrl $diskEncryptionKeyVaultUrl -DiskEncryptionKeyVaultId $KeyVaultResourceId -VolumeType $VolumeType -SequenceVersion $sequenceVersion -skipVmBackup -EncryptFormatAll

* 1. Check encryption status. You can check status once every 10 minutes, the devices will be encrypted in minutes even if they have a big size since we're using EncryptFormatAll
  2. Wait for the completion status to get into “Encryption succeeded for data volumes” state in the "progressMessage" field.
* PowerShell

Get-AzVmDiskEncryptionStatus -ResourceGroupName $rgName -VMName $vmName

* Azure CLI

az vm encryption show --name $vmName --resource-group $rgName --query '[status,substatus]'

* 1. You can validate the data disks are mounted (/data1 and /data2) and encrypted with the below commands

Lsblk -f

df -h /data\*

* 1. You can notice the file systems were added to /var/lib/azure\_disk\_encryption\_config/azure\_crypt\_mount (in case of an old encrypion) or added to /etc/crypttab file in case or a newer encryption.
     + Do not modify any of these files.
     + This is going to be the file that will be taking care of activating these disks during the boot process so they can be later used by LVM. Do not worry about the /data0 or /data1 mount points, as ADE will lose the ability to get the disks mounted as a normal file system after we do a pvcreate on top of those encrypted devices (which will get rid of the file system format we used during the preparation process).

~~cat /var/lib/azure\_disk\_encryption\_config/azure\_crypt\_mount~~

cat /etc/crypttab

* 1. Ensure the encrypted LUNs are unmounted so they become available and be used by LVM in the next steps.

umount /data0

umount /data1

# Creating LVM structures on top of the encrypted layer(s)

1. Now the underlying disks are encrypted you can proceed to create the LVM structures. Instead of using the device name, use the /dev/mapper paths for each of the disks to perform a pvcreate (on the crypt layer on top of the disk not on the disk itself).

fdisk -l | grep mapper

more /etc/crypttab

1. Create a Volume group called vgdata(or any other desired name) with 2 volumes lvdata0 and lvdata1 that will be mounted as /data0 and /data1 respectively.
   1. Initialize the PVs, at this moment the encrypted layer will be wiped

echo "y" | pvcreate /dev/mapper/~~76e57a92-7acd-4357-a15f-dde9bb1d459d~~

echo "y" | pvcreate /dev/mapper/~~33a0266c-e98e-48c0-8711-4d7b98b86092~~

Note: Replace the dev mappers (/dev/mapper/76e57a92-7acd-4357-a15f-dde9bb1d459d and /dev/mapper/33a0266c-e98e-48c0-8711-4d7b98b86092) accordingly. You can get those from the /var/lib/azure\_disk\_encryption\_config/azure\_crypt\_mount file, the /etc/crypttab file or from the lsblk command output

You will get a warning asking if it is OK to wipe out the filesystem signature. You may proceed by entering ‘y’ or simply use the echo “y” as shown in the command above.

* 1. Create the VG using the same devices already initialized

vgcreate vgdata /dev/mapper/76e57a92-7acd-4357-a15f-dde9bb1d459d /dev/mapper/33a0266c-e98e-48c0-8711-4d7b98b86092

vgdisplay -v vgdata

Note: Replace the dev mappers (/dev/mapper/76e57a92-7acd-4357-a15f-dde9bb1d459d and /dev/mapper/33a0266c-e98e-48c0-8711-4d7b98b86092) accordingly. You can get those from the /var/lib/azure\_disk\_encryption\_config/azure\_crypt\_mount file, the /etc/crypttab file or from the lsblk command output

* 1. Create two LVM volumes. In this example we are creating 2 volumes of 4GBs each, they are named lvdata0 and lvdata1

lvcreate -L 4G -n lvdata0 vgdata

lvcreate -L 4G -n lvdata1 vgdata

1. Create the file systems over the LVM volumes, in this case ext4 it’s used, you can also use XFS if needed:

echo "yes" | mkfs.ext4 /dev/vgdata/lvdata0

echo "yes" | mkfs.ext4 /dev/vgdata/lvdata1

1. Add the new the file systems to /etc/fstab

echo "/dev/mapper/vgdata-lvdata0 /data0 ext4 defaults,nofail 0 0" >>/etc/fstab

echo "/dev/mapper/vgdata-lvdata1 /data1 ext4 defaults,nofail 0 0" >>/etc/fstab

mount -a

Important: please make sure the nofail option is added to the mount point options of the LVM volumes created on top of an ADE encrypted device. This is very important to avoid the OS from getting stuck during the boot process (or in maintenance mode). The encrypted disk will be unlocked at the end of the boot process and the LVM volumes and file systems will be automatically mounted until they are unlocked by ADE, if the nofail option is not used, the OS will never get into the stage where ADE is started and the data disks are unlocked and mounted.

1. Check the file systems are getting mounted

df -h

1. You can test rebooting the VM and validating the file systems are also automatically getting mounted after boot time. Please take under consideration that this process may take several minutes depending of the amount of file systems and the sizes

reboot

df -h /data\*

# Reference

* [Azure Disk Encryption for Linux VMs](https://docs.microsoft.com/en-us/azure/virtual-machines/linux/disk-encryption-overview#--use-the-encryptformatall-parameter-with-logical-volume-manager-lvm)