Block device encryption and zuluCrypt.

ZuluCrypt is a simple, feature rich and powerful solution for hard drives encryption for linux.

ZuluCrypt supports LUKS, TrueCrypt, VeraCrypt and PLAIN dm-crypt encrypted volumes.

These supported encrypted volumes may resides in image files, hard drives and usb sticks, LVM volumes as well as in mdraid devices.

There are two kinds of encrypted volumes. Those that use what is commonly know as “a header” and those that do not. TrueCrypt, VeraCry and LUKS volumes use a header. PLAIN dm-crypt volumes do not use a header.

There are two kinds of header using encrypted volumes. Those that use an encrypted header and those that do not. TrueCrypt and VeraCrypt use an encrypted header whereas LUKS does not. The use of non encrypted header in LUKS makes it obvious to everybody that the volume is an encrypted LUKS volume and this may be problematic to some people. How big the problem may be depends on the person and their use case for hard drive encryption.

The use of encrypted header as in TrueCrypt or VeraCrypt volumes or no header at all as in PLAIN dm-crypt volumes make these volumes indistinguishable from random noise and this may seem useful at a glance but its usefulness may not hold up against scrutiny as the likelihood of being believed that a 100GB file made up of cryptographically sound random data is just a 100GB file made up of random data and not a container file for an encrypted volume is not very high.

With LUKS, TrueCrypt and VeraCrypt volumes, it is very important to have a volume header backup since a valid header is required to unlock the volume. A corrupted or missing header will make the volume unusable causing the loss of all encrypted data.

LUKS stands for “Linux Unified Key Setup”. It is a specification of how to store information necessary to open a LUKS formatted encryption volume. LUKS encryption format is the standard format in linux and a recommended one if the encrypted volume is to be used among linux systems. TrueCrypt or VeraCrypt volumes are better alternative if the encrypted volume is to be shared between linux, windows and OSX computers.

ZuluCrypt can create and open 5 types of encrypted volumes, LUKS, TrueCrypt, VeraCrypt,PLAIN dm-crypt and PLAIN dm-crypt volume at a none zero offset. PLAIN dmcrypt volume is a header less encrypted volume and all necessary encryption information is provided by zuluCrypt when it creates or open these volumes.

Pros and cons of the five volumes.

PLAIN dm-crypt:

Pro:
1. It does not use a volume header and hence its not possible to “brick” the entire volume simply by over writing a small part of it.
2. It does not use a header and hence its impossible to know if the volume is made up of only

cryptographically sound random data or if its an encrypted volume.

Cons:
It does not use a header and hence any tool that opens these volumes must provide the encryption options that were used when the volume was created. Different tools may use different encryption options making these encrypted volumes not very portable between applications or even between different versions of the same application.

PLAIN dm-crypt at a none zero offset.

This volume has the same pros and cons as those of a PLAIN dm-crypt volume.

Additional pro for this volume is that it can be places anywhere on the device making it possible to have this volume on top of any one of the other supported encrypted volume or on top of an unencrypted volume. For example, its possible to have an X MB drive that has unencrypted file system at the beginning of the device and a "PLAIN dm-crypt volume with an offset" somewhere towards the end of the device making it possible to use the drive as unencrypted volume and as encrypted volume simultaneously depending on sensitivity of the data to be stored on the device. If this volume type is to be used, preceding part of the drive should be formatted in "FAT" family of file systems.

This volume types gives a "hidden volume" type functionality offered by TrueCrypt and VeraCrypt.

When creating or unlocking this volume type, the starting offset of the volume will be asked and NOT the volume size as TrueCrypt and VeraCrypt does with the hidden volume.

The above means, if you have a 100 MB drive and you want to create a 30MB "PLAIN dm-crypt volume at a none zero offset", you will enter the starting offset of the volume as 70MB. In VeraCrypt or TrueCrypt, you will enter the hidden volume size of 30MB. The starting offset and the size of the hidden volume are related by a simple formula: starting offset(70MB) = device size(100MB) – hidden volume size(30MB).

TrueCrypt
Pro:
1. It uses an encrypted header and hence its not possible to know if the volume is TrueCrypt formatted encrypted volume or if the volume is just made up of cryptographically sound random data.

2. Hidden volume. A TrueCrypt volume can have up to two different encrypted volumes. The first volume is commonly know as "outer volume" and the second optional one is commonly known as "hidden volume".When a TrueCrypt volume is about to be opened, the user has an option to select which one of the two to open by giving appropriate key.

Cons:
1. It uses an encrypted header and it is not possible to open the volume without a valid header. If you use a TrueCrypt volume, make sure you have at least one backup of the volume header.

VeraCrypt
VeraCrypt is an extension of TrueCrypt and it shares the same TrueCrypt's pros and cons.

Additional Pro for VeraCrypt over TrueCrypt.

1. It requires stronger effort to unlock VeraCrypt volume and this makes them more secure over TrueCrypt volumes.

Additional Cons for VeraCrypt over TrueCrypt.
1. It requires stronger effort to unlock a VeraCrypt volume and this increases the time it takes to unlock a VeraCrypt volume. How long it will take depends on the strength of the computer and it may vary from a few seconds to several minutes.

<u>LUKS</u>
Pro:
1. A LUKS volume can be opened with up to 8 different keys.

Cons:
1. A LUKS header is stored unencrypted making it obvious the volume is LUKS formatted encrypted volume and this may not be desirable under certain circumstances. It is possible to create a LUKS volume with a detached header and zuluCrypt can open these volumes using “luks” plugin.
2. It uses a header. As it is not possible to open a header using encrypted volume without its header, a corrupted LUKS header makes it impossible to open the volume. If you use a LUKS volume, make sure you have at least one backup of the volume header.

ZuluCrypt can do two types of encryption. It can do single file encryption/decryption or block device encryption.

<u>File encryption.</u>

ZuluCrypt can encrypt and decrypt individual files. This feature is useful when a user just wants to encrypt a single file and taking the route of creating an encrypted container file to host the file is seen as an unnecessary hassle. This functionality is akin to file encryption using gpg with a symmetric key.

File encryption is done using libgcrypt as a cryptographic backend. Files are encrypted using 256 bit AES in CBC mode. The encryption key is derived from user pass phrase using pbkdf2 with 10,000 rounds of iterations and sha2 as a cryptographic hash function. The resulting encrypted file will have a file size that equals (64 + 1024 * n) bytes where n is a number starting from zero.

How to create an encrypted file:

1. Start zuluCrypt.
2. Go to the menu and then click “zC->encrypt a file” to open a file encryption dialog window.
3. At the dialog that will show up, click the button that is on the same line as “source path” text. A file dialog will show up, select the file you want to store encrypted, enter the password to be used to encrypt the file and then click “create” and the encrypted version of the file will be created at the path given by “destination path” field.

To decrypt the file created with above steps:
1. Start zuluCrypt.
2. Go to the menu and then click “zC->decrypt a file” to open a file decryption dialog window.
3. At the dialog that will show up, click the button that is on the same line as “source path” text. A file dialog will show up, select the file you want to decrypt, enter the password to be used to decrypt the

file and then click “create” and the decrypted version of the file will be created at the path given by “destination path” field.

<u>Block device encryption.</u>

A hard drive or a usb stick are two examples of block devices. A regular file can simulate a block device through a use of devices known as “loop devices”. These devices have a device path that starts with “/dev/loop”.

The infrastructure in the linux kernel that deal with block device encryption is called “dm-crypt” and it does its work through a process commonly known as OTF(on the file encryption). Dm-crypt devices are represented by device addresses that starts with “/dev/dm-” and these paths are usually accessed through their soft links that reside in “/dev/mapper”.

Below is an example of steps taken in creating a 100MB encrypted container in a file and adding a file in it to be stored securely.

1. Create a 100MB file.
2. Attach a loop device to the file.
3. Create an OTF encryption mapper against the loop device.
4. Put a file system on the encryption mapper.
5. Mount the file system on the mapper.
6. Copy The file to be stored securely to the file system through the mount point.
7. Unmount the file system.
8. Destroy the OTF encryption mapper.
9. Detach the loop device from the file.
10. Maintain the encrypted volume as a secure holder of files within it.

All zuluCrypt does is provide a GUI to make it easy to do above specified tasks.

With the above steps:
Step 1 deal with a path that look like “/home/ink/secret.img”, this is a path to a regular file.

Step 2 converts “/home/ink/secret.img” file to something like “/dev/loop0” loop device path.

Step 3 converts “/dev/loop0” loop device path to something like “/dev/mapper/secrets.img”. Data written to “/dev/mapper/secrets.img” will get encrypted and then passed forward to “/dev/loop0” on its way to “/home/ink/secret.img”. When data is read from “/dev/mapper/secrets.img”, the data will be read from “/dev/loop0” who in turn will read it from “/home/ink/secret.img”, decrypted by dm-crypt and then given to the reader. This process is called “on the fly encryption” because the encryption mapper does not store or hold on to data, it gets data and then encrypts or decrypts it depending on the direction of data flow and then passes it along.

<u>How to create an encrypted container in an image file.</u>

1. Start zuluCrypt.
2. Go to “menu->create->encrypted container in a file” to open a dialog window.
3. Enter the name of the file to be used to hold the container in the “file name” field.
4. Enter the size of the container in the “file size” field.

5. Click “create”.
6. Wait for the container file to be created and for the volume creation dialog to show up.
7. Enter the password to be used to create the volume.
8. Select the type of volume you want to create from the “volume type” list.
9. Click create to create the volume.

<u>How to create an encrypted container in a partition.</u>

1. Start zuluCrypt.
2. Go to “menu->create->encrypted container in a hard drive” to open a dialog window.
3. Click/double click on the hard drive you want to create a volume in and then advance to instruction number 7 in the instruction list above. If the partition you want to put an encrypted container does not show up on the list, then restart zuluCrypt from root's account and try again.

<u>How to open an encrypted container that reside in a file using zuluCrypt.</u>

1. Start zuluCrypt.
2. Go to “menu->open->encrypted container in a file” to bring up a dialog window.
3. On the dialog window, click the button to the right of “volume path” field and then browse to where the volume is and click it to open it. Alternatively, you can just drag the volume file on zuluCrypt to generate a password dialog prompt with the file path already filled in.
4. Enter the volume key in the volume key field and then click “open” to open the volume.

<u>How to open an encrypted container that reside in a partition using zuluCrypt.</u>

1. Start zuluCrypt.
2. Go to “menu->open->encrypted container in a partition” to bring up a dialog window.
3. On the dialog window, click/double click on the partition with an encrypted volume you want to open.
4. Enter the volume key in the volume key field and then click “open” to open the volume.

With both two steps above, the volume will be opened and mounted at a path whose last component is given by the entry in the field “mount name”. When the volume is successfully opened, zuluCrypt will automatically open the mount point path. To close the volume, click its entry on the zuluCrypt window and then click “close” on the pop up window.

ZuluCrypt can open an encrypted volume using keys derived from different sources. These sources include, a pass phrase, a key file, a key retrieved from kwallet, a key retrieved from Gnome's libsecret, a key retrieved from an internal secure storage system, a key from gpg encrypted key file among other sources.

To use a pass phrase volume key, make sure the key source option read “key” and then enter the pass phrase on the entry field at the bottom.

To use a keyfile as the source of volume key, click the option bar and then select “keyfile” and then press the button on the lower right to bring a dialog box that will allow you to browse to where the key file is.

To use a plugin as the source of volume key, click the option bar and then select “plugin” and then press the button on the lower right to bring up a list of available plugins and then select the one you want from the list.

Volume keys stored in kwallet, Gnome keyring or internal secure storage system plugins can be managed by going to “menu->options->manage volumes in internal/kde/gnome wallet”.

Storage of keys in a gnome wallet/keyring seem most appropriate in a gnome session but this has some security repercussions, the keys are stored in the user keyring and this keyring gets unlocked when the user logs in. This means that once a user is logged in and the keyring is open, any application that runs in that user session can read those keys using public APIs exposed by the storage system.

In a kde system, a kwallet secure storage system seem most appropriate but it suffers from the same security problem the gnome secure storage system has, once the wallet is open, any application running in the user session can access it using public APIs exposed by the storage system.

The behaviors of the above secure storage systems is by design but this design may not be ideal for some users under certain use cases. The internal secure storage system is powered by libgcrypt and it does not have the behavior of the above two systems. An unlocked internal secured storage system is accessible only to the instance of zuluCrypt that unlocked it.

Favorites.

For convenience, most used volumes can be easily opened by adding them to the favorite list. Entries on the list are added in the dialog window opened by clicking “menu->options->manage favorites”. Favorite entries are added by clicking the “favorite” entry on the menu.

Erase data in a device.

It is very important to create encrypted volume over cryptographically strong random data to make it impossible to know what part of the encrypted volume has been used and what part has not. If the encrypted volume is created over predictable data patterns like on a device with only zeros in it, forensic analysis may reveal how much and what part of the encrypted volume are in use.

When creating an encrypted container in a device, zuluCrypt offers an option to first write random data over the device. This feature can be performed on other devices by activating it through “menu->erase data in a device”. Random data are written to disk by opening a plain dm-crypt encryption mapper on the device with a 64 byte random key and then blasting zeros on the device through the mapper. This technique has proven to be faster compared to alternatives like writing random data on the device read from “/dev/urandom”.

System and non system volumes.

To enforce access controls on what user can access what block device and what they can do with the access they have, zuluCrypt employes a concept of “system volumes” and “non system volumes”.

A system volume is defined as a volume that has an active entry in “/etc/fstab”,”/etc/crypptab”,“/etc/zuluCrypt/system_volumes.list” or if udev identify it as such if udev is enabled. Ideally, all volumes inside the computer are to be considers system volumes.

A non system volume is a volume that failed in the above considerations or if it has an entry in “/etc/zuluCrypt/non_system_volumes.list”. Ideally, these volumes are plug gable usb based hard drives or usb sticks.

Partitions can be added or removed from the list of system or non system volumes simply by starting zuluCrypt from root's account and then going to “menu->options->manage system volumes/manage non system volumes” and then adding the volume in the appropriate list.

Permissions.

ZuluCrypt limits what a user can do on block devices through unix's group based permission system using two groups, “zulucrypt” and “zulumount”.

If a device is identified as a system device, only a root user or a user who is a member of group “zulucrypt” can create an encrypted volume in the device or taking/restoring volume headers. If you want to create a volume in a device and the device does not show up on the list, restart zuluCrypt from root's account and try again.

If a device is identified as a system device, zuluMount will mount it only if the user is root, is a member of group “zulumount” or the device has an entry in “/etc/fstab” with either “user” or “users” mount options set.

ZuluMount.

ZuluMount is a general purpose mounting tool that can open zuluCrypt supported encrypted volumes as well as non encrypted volumes.

ZuluMount can also auto detect plugged in devices and auto mount them.

ZuluMount can also unlock encfs volumes.

Copyright (c) 2015-2016 Francis Banyikwa, mhogomchungu@gmail.com