Linux on z Systems and LinuxONE

Getting started with pervasive disk encryption

September 2017

SC34-2783-00

Pervasive disk encryption

Pervasive disk encryption forms the foundation for pervasive encryption as introduced with IBM® z14™ (z14). With z14 hardware, you can protect data-at-rest against unauthorized access while balancing the constraints of implementation complexity, cost, and system performance.

This publication describes the steps for setting up encrypted data volumes. This setup ensures that the keys used to encrypt the data cannot be stolen and used for offline attacks.

With this setup, you first generate secure keys with IBM cryptographic coprocessors. Then you efficiently use protected keys converted from the secure keys to encrypt and decrypt data transparently to applications. The protected key cryptography is supported by the CP Assist for Cryptographic Functions (CPACF). Applications do not require any changes to use the encrypted volumes.

Prerequisites

These hardware and software components are required for implementing pervasive disk encryption.

Hardware prerequisites

- An IBM z14 or z13™ with the CPACF feature installed. The use of the CPACF requires the appropriate microcode to be loaded which you can order as no-charge feature code (LIC #3863).
- For redundancy, two IBM Crypto Express5 or Crypto Express6 adapters in CCA coprocessor mode (CEX5C or CEX6C).
- A Trusted Key Entry (TKE) workstation. For non-production environments you can use the utilities from the CCA package instead of the TKE to set master keys.
- SCSI or DASD volumes to be encrypted.

Software Prerequisites

- Any Linux distribution that includes support for the pkey and paes_s390 kernel modules. Linux kernel 4.12 or later includes the required kernel modules.
- The cryptsetup utility used to configure an encrypted volume.
- The **zkey** utility from the s390-tools package (version 1.39 or later). Use this utility to generate and manage secure keys.
- The CCA package to connect the TKE workstation to the cryptographic coprocessors.

Assumptions

- A Linux instance is installed as a z/VM® guest or in an LPAR.
- Two cryptographic coprocessor domains are configured to the LPAR or as dedicated adapters to the z/VM guest.
  - The two domains have the same domain ID and are loaded on two distinct cryptographic coprocessors.
  - The same AES master key is set for the domains of both cryptographic coprocessors.

- The volumes to be encrypted are configured to be persistently available to the Linux instance.

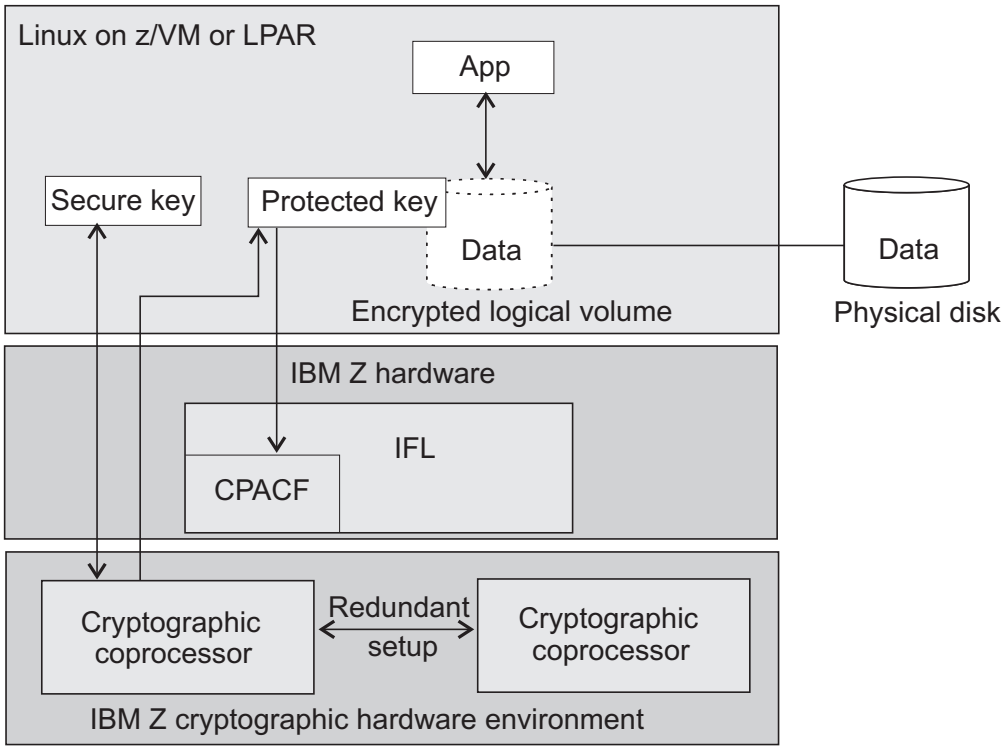

Figure 1. System configuration for pervasive disk encryption

The main characteristics of pervasive disk encryption are:

- An enhanced security achieved by the secure key operations and management in the tamper-proof cryptographic coprocessors.
- A performance advantage of protected key cryptography compared to secure key cryptography. Therefore, the secure key is converted into a protected key. Protected key cryptography is running at processor speed in the CPACF.

Creating and accessing an encrypted partition

Setting up disk encryption entails generating secure keys and creating logical volumes.

Before you begin

You require free disk partitions that are configured to be persistently available to your Linux instance. Make sure that the pkey and paes_s390 modules are loaded into the kernel.

Based on the sample system environment as shown in Figure 1 on page 2, the procedure documented here uses the first partition on a multipath SCSI disk: `/dev/mapper/mpathb-part1`

Procedure

1. Use the **zkey** utility to generate a secure key in a file. For example, issue the following command for the recommended XTS cipher mode:

   ```
   # zkey generate /etc/secure_keys/xts-secure-key.sk --xts
   ```

   In the example, the generated secure key file is stored in the `/etc/secure_keys/` directory.

   You can have a secure key per volume or share a secure key among volumes.
2. Use the **cryptsetup** utility to create an encrypted logical volume. Use the plainOpen function to open (unlock) the partition and assign a logical volume name. This function creates a logical volume in `/dev/mapper`.

   When you access a partition, you need to specify:

   - The location and name of the secure key file.
   - The key size (in bits). For XTS, the key size is 1024.
   - The paes cipher and its operation mode (in the example, XTS).
   - The name of the partition.
   - A name of your choice for the logical volume.

   For example, if the name of the partition is `mpathb-part1`, and the assigned logical volume name is `enc-data1`, then you invoke the **cryptsetup** utility as follows:

   ```
   # cryptsetup plainOpen --key-file /etc/secure_keys/xts-secure-key.sk --key-size 1024 \
                          --cipher paes-xts-plain64 /dev/mapper/mpathb-part1 enc-data1
   ```

   You can check the result of this step with the command **ls /dev/mapper/**. Any I/O operation to or from `/dev/mapper/enc-data1` will then be transparently encrypted or decrypted onto the `/dev/mapper/mpathb-part1` partition. As of now, do not write to this partition directly.
3. Unlock the volume during the boot process. Create an entry in `/etc/crypttab` to persistently configure an unlocking at boot time. Each line describes an encrypted volume and assigns the secure key to be used for encryption and decryption of the partition:

```
# /etc/crypttab
#
# See crypttab(5) for more information.
#
#
# Target  Source  device            Key file                          Options
enc-data1 /dev/mapper/mpathb-part1 /etc/secure_keys/xts-secure-key.sk cipher=paes-xts-plain64,size=1024,hash=plain
```

The format of the `/etc/crypttab` file depends on your Linux distribution. See the `crypttab` man page for more details.

What to do next

Once you have opened an encrypted logical volume either with the **cryptsetup** utility (step 2 on page 3), or implicitly during the boot process (step 3 on page 3), you can use this volume like any other block device. Typical next steps are:

- If you want to manage your encrypted disks using LVM, create LVM physical volumes and add them to an LVM volume group.
- Create a file system on the encrypted logical volume.
- Create a mount point and update `/etc/fstab` to later mount the file system on the encrypted logical volume or LVM logical volume.

SC34-2783-00