

Encrypted Storage Technology

Travis H.

travis+security@subspacefield.org

<http://www.subspacefield.org/~travis/>

XCSSA, 18 Feb 2008

Outline

- 1 Encrypted Storage Technologies
 - Why You Care
 - Three Generations
- 2 Application File Encryption
- 3 Filesystem Encryption
 - Overview
 - CFS
 - TCFS
- 4 Block Device Encryption
 - Overview
 - LUKS (Linux Unified Key Setup)
 - dm-crypt
 - OpenBSD's svnd
 - FreeBSD Disk Encryption

Why You Care

- Theft, loss, or confiscation[7]
- Drive failure and disposal
- Fire, hurricane, flood, or any other disaster
- Owner runs short on funds and wants to liquidate some assets quickly
- Secure deletion not guaranteed to work, never fast
- Encrypted storage *secure by default*

Three Generations

- 1 Files: GPG, PGP
- 2 Filesystems: CFS (Unix), TCFS (BSD), EFS (Microsoft)
- 3 Block Devices: PGPDisk (Microsoft), TrueCrypt, dm-crypt (Linux), svnd (OpenBSD)

Three Generations

- 1 Files: GPG, PGP
- 2 Filesystems: CFS (Unix), TCFS (BSD), EFS (Microsoft)
- 3 Block Devices: PGPDisk (Microsoft), TrueCrypt, dm-crypt (Linux), svnd (OpenBSD)

Three Generations

- 1 Files: GPG, PGP
- 2 Filesystems: CFS (Unix), TCFS (BSD), EFS (Microsoft)
- 3 Block Devices: PGPDisk (Microsoft), TrueCrypt, dm-crypt (Linux), svnd (OpenBSD)

Application File Encryption

First Generation: Application Layer File Encryption

- Easiest to implement - no OS support required
- May be a simple Unix filter
- Hard to use - requires user to do all the work to remain secure, prone to pilot error
- Hard to implement safely - plaintext may be paged out, or written to /tmp
- Examples: PGP, GPG, mdecrypt, bccrypt, cccrypt

Filesystem Encryption

Second Generation: Filesystem Encryption

- Attempt to automate the encryption of certain files (usually all under a certain mount point)
- Requires knowledge of files and directories, so has to act as a file system to OS
- Thus, OS-specific and surprisingly complex - file system API (“cross-section”) too large
- Where do you hide all the metadata like IVs and keys?

CFS

CFS (Crypting File System)

- Written by Matt Blaze, security guru
- First attempt at encrypted filesystem, pieces go back to 1987!
- Pretends to be NFS server to clients (often localhost), stores encrypted data on local file system (or remote NFS server)
- Somewhat buggy, hard to access internals
- Messy, leaves dangling symlinks all over (to store IVs)
- I accidentally corrupted my system (basically lost the IVs) but was able to use classical cryptanalysis techniques to recover the data[5]
- Deprecated

TCFS

TCFS (Transparent Crypting File System)

- Written by some guys in Italy
- Used BSD stackable file system technology to implement an encryption layer “transparently” on top of any other file system
- Turns out to be surprisingly hard, because you want to hide all the metadata; stored the IV at the beginning of each file, but then had to do “fixup” on file sizes.
- Will work on Linux with kernel patches[6]
- Deprecated, author email addresses don't work

Block Device Encryption

Third Generation: Encrypted Block Devices

- Appears as a block device (e.g. disk) so very simple API; just encrypts or decrypts blocks of data and writes to underlying store.
- Underlying store is often disk partitions, but sometimes can use files on a different filesystem.
- Usually hides metadata in beginning of area, simply adjusts size of plaintext device to hide it
- Separation of duty; can be used with any filesystem

LUKS

- Standardizes on-disk format for encrypted contents and metadata[2]
- Implements TKS1 key format[4], which supports:
- Passphrase revocation (without re-encrypting)
- Multiple passphrases
- Protects against dictionary attacks via PKCS#5 PBKDF2

dm-crypt

dm-crypt (device-mapper crypto)

- The new hotness for the Linux kernel
- Available *by default* in the debian installer and “alternate” ubuntu install CD
- Now you can encrypt everything but `/boot`!
- Linux kernel facility but made available to userland via `cryptsetup` binary
- Highly recommended

LUKS-compliant cryptsetup

- 1 `cryptsetup luksFormat /dev/sda`
(prompts for passphrase, can specify a file containing phrase instead)
- 2 `cryptsetup luksOpen /dev/sda crypted`
- 3 `mkfs [...] /dev/mapper/crypted`
- 4 `mount /dev/mapper/crypted /crypt`

If you want to encrypt the operating system, you really should use a distro that supports it, because it's rather difficult to add to the boot process.

OpenBSD's svnd

- vnode disk driver allows you to make a file appear as a disk
- secure variant does encryption at same time
- available to userland through vnconfig
- can't encrypt OS yet
- *OpenBSD swap encrypted with random key by default*
- Protects against dictionary attacks via PKCS#5 PBKDF2

OpenBSD's vnconfig

- 1 `dd if=/dev/random of=/etc/seed bs=1k count=1`
- 2 `vnconfig -K 1000 -S /etc/seed svnd0 /etc/encrypted_disk`
combines passphrase with seed 1000 times
- 3 `newfs /dev/svnd0c`
- 4 `mount -o nodev,nosuid /dev/svnd0c /crypt`

FreeBSD Disk Encryption

- Geometry-Based Disk Encryption
- Requires loading kernel module or compiling into kernel
- Can't encrypt OS yet (no support in bootloader)
- Two flavors, GBDE and GELI
- GELI supports PKCS#5 PBKDF2

Using FreeBSD GBDE

- 1 `kldload geom_bde` (or add to kernel config and recompile)
- 2 `mkdir /etc/gbde`
- 3 `gbde init /dev/ad4s1c -i -L /etc/gbde/ad4s1c.lock`
- 4 `gbde attach /dev/ad4s1c -l /etc/gbde/ad4s1c.lock`
- 5 `newfs -U -O2 /dev/ad4s1c.bde`
- 6 `mount /dev/ad4s1c.bde /crypt`

Using FreeBSD GELI

- 1 `geom_eli_load="YES" in /etc/bootloader.conf`
- 2 `dd if=/dev/random of=/root/da2.key bs=64 count=1`
- 3 `geli init -s 4096 -K /root/da2.key /dev/da2`
- 4 `geli attach -k /root/da2.key /dev/da2`
- 5 `newfs /dev/da2.eli`
- 6 `mount /dev/da2.eli /crypt`

For Further Reading I



Travis H.

Security Concepts

<http://www.subspacefield.org/security/>



<http://luks.endorphin.org/>



TrueCrypt homepage

<http://www.truecrypt.org/>



TKS1

<http://clemens.endorphin.org/TKS1-draft.pdf>



Travis H.

CFS travails

http://www.subspacefield.org/~travis/cfs_travails.txt

For Further Reading II



TCFS article in LINUX Journal

<http://www.linuxjournal.com/article/2174>



Schneier

How to Secure Your Computer, Disks, and Portable Drives

http://www.schneier.com/blog/archives/2007/12/how_to_sec