Comparação Entre Dois Sistemas de Arquivos Criptográficos

Jessica Imlau Dagostini * Marisa Richter †

7 de julho de 2017

Resumo

A necessidade de manter dados seguros é cada vez maior. Para isso, o uso de sistemas de arquivos criptográficos pode ser um aliado nessa segurança. Há a possibilidade de usarmos estes sistemas de arquivos tanto em *kernelspace* quanto em *userspace*. O presente trabalho visa comparar dois *ecrypted file systems*, sendo um de cada espaço, a fim de auxiliar na escolha.

Palavras-chaves: sistemas de arquivos. criptografia. linux.

Abstract

Keep data safe is a necessity that keeps growing. Use encrypted file systems can be an ally in this security. There are possibilities to use these file systems in *kernelspace* or *userspace*. This paper aims to compare two *encrypted filesystems*, being one from each space, and helps to choose which option uses in real cases.

Keywords: file systems. cryptography. linux.

1 Introdução

Manter seguros dados, arquivos ou quaisquer outras informações em meio digital virou uma das principais preocupações da atualidade. Depois do advento da internet, a alta comunicação e acesso a dados além de trazer benefícios, também trouxe malefícios. Toda informação online precisa de diversos níveis de proteção, desde o nível de proteção de rede (através de firewalls) até nível de proteção dos arquivos a serem acessados. Para este último fim, uma forma de proteção vêm sendo pensado a fim de fortificar a segurança: sistemas de arquivos c riptográficos.

Neles a organização dos arquivos possui uma camada extra, que lida com a criptografia dos arquivos que ele gerencia. Há tecnologias que lidam com a criptografia em todo o disco, usando o driver do dispositivo para criptografar e descriptografar arquivos.

*jessicadagostini@gmail.com
†marisaengc@gmail.com

Outras tecnologias operam em cada arquivo, geralmente necessitando de operações de reescrita sobre os arquivos, a fim de possibilitar um uso transparente da criptografia. Por fim, também há a abordagem que criptografa todo o sistema de arquivos, permitindo a manipulação de criptografia em cada arquivo ou diretório do sistema usando uma única chave (SYMANTEC, 2003).

Sobre esta última abordagem, estes sistemas de arquivos podem ser manipulados em modo *kernel* ou em modo usuário. Para conseguir ser manipulado em modo usuário, faz-se uso de uma interface para programas de *userspace* exportarem um sistema de arquivos para o *kernel* do Linux, chamada *libfuse*. Um FUSE (*Filesystem in userspace*, sistema de arquivos de espaço de usuário) é uma aplicação que usa a *libfuse* para enviar requisições ao *kernel* (LIBFUSE, 2016).

O presente artigo fará um estudo e comparação entre dois EFS, *Encrypted File Systems* (Sistemas de Arquivos Encriptados), um em *kernelspace* (eCryptFS) (HALCROW, 2005) e outro em *userspace* (EncFS) (GOUGH, 2003), ambos suportados por sistemas Linux. A seção a seguir abordadará a estrutura de criptografia de arquivos em modo núcleo, fazendo especial introdução e explanação sobre o funcionamento do *eCryptFS*. Na terceira seção será feito o mesmo sobre criptografia de arquivos em modo usuário e o *EncFS*. Em sequência serão comparadas as duas formas de criptografia em diferentes espaços, tendo como base as informações obtidas através da bibliografia e também de testes práticos reais feitos pelas autoras do trabalho. Por fim, serão apresentas considerações finais sobre o presente trabalho e suas respectivas referências.

2 Sistemas de arquivos em *kernel space*

Quando implementado em modo *kernel*, um sistema de arquivos geralmente possibilita uma melhor performance visto que o número de trocas de contexto é bem reduzida se comparada à aplicações que não possuem privilégios de acesso. Além de um melhor desempenho, sistemas de arquivo em modo núcleo oferecem maior segurança uma vez que as informações contidas no *kernel* não são tão fáceis de serem acessadas por usuários maliciosos. Por todos estes fatores, implementar um sistema de arquivos que possibilite uma encriptação automática de todo seu repositório torna o dispositivo mais seguro.

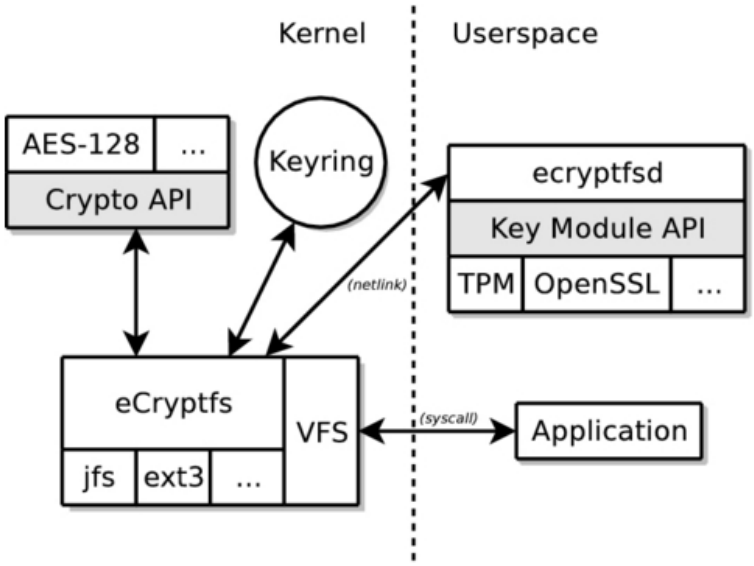
2.1 eCryptFS

O *eCryptFS* é um sistema de arquivos criptográfico que estende o *CryptFS* (ZADOK; BADULESCU; SHENDER, 1998) e pode ser montado em um diretório ou sob qualquer outro sistema de arquivos, como UFS e NFS. Por ser implementado em modo núcleo, oferece algumas vantagens e melhores desempenhos. (HALCROW, 2005)

O *eCryptFS* é nativo para o *kernel* do Linux e para poder utilizá-lo é preciso ter uma ferramenta no espaço de usuário chamada *ecryptfs-utils*, que por usa vez precisa da *keyutils*, ferramentas estas necessárias para o sistema de gerenciamento de chaves do *kernel*.

O seu funcionamento, conforme ilustrado pela figura 1, é basicamente uma requisição a dados da aplicação para o Sistema Virtual de Arquivos do *kernel* (VSF).O *eCryptFS* recupera a chave da sessão do usuário e utiliza uma API (*Application Programming Interface*) criptográfica, um anel de chaves do *kernel* e um emulador do *eCryptFS* no espaço de usuário para realizar a criptografia e descriptografia do conteúdo dos arquivos. (HALCROW, 2007)

Figura 1 – Funcionamento do *eCryptFS*



Fonte: *eCryptfs: a Stacked Cryptographic Filesystem.* Disponível em: <http://www.linuxjournal.com/article/9400> Acesso em Jul. 2017

.

Como o *eCryptFS* tem o objetivo de ser flexível na manipulação das chaves, como um *Pretty Good Privacy*[1] (PGP) tendo base nas especificações *OpenPGP*[2], um usuário gera duas chaves, uma pública e uma privada. A chave pública utilizada para criptografar os dados, e a chave privada correspondente poderá descriptografar os mesmos.

3 Sistemas de arquivos em *user space*

Também conhecidos como FUSE (*Filesystem in Userspace*), sistemas de arquivos em espaço de usuário são aqueles em que os dados e os metadados são produzidos por um processo de usuário e exportados para o modo *kernel* utilizando o módulo FUSE do núcleo e da biblioteca libfuse (RAWAT; KUMAR, 2012) (LIBFUSE, 2016). Essa biblioteca fornece funções de montagem e desmontagem de *file systems*, leitura de requisições e envio de respostas ao *kernel*. Sua principal característica é a possibilidade de criação de sistemas de arquivos seguros por usuários sem privilégios (*non-root users*). Estas organizações são de domínio único do usuário que a criou (KERNEL.ORG, 2017). Esta biblioteca proporciona uma boa forma de escrever *virtual file systems*.

3.1 EncFS

O *EncFS* é um sistema de arquivo criptografado *FUSE-based* que visa proteger os dados com mínimos problemas (ARCHLINUX, 2014). Ele usa diretórios definidos pelo usuário para armazenar os arquivos criptografados, sendo necessários dois arquivos na montagem do sistema: o diretório fonte, que contém os arquivos criptografados; e o ponto

[1] Programa de encriptação e descriptação de dados, com chave pública, fornece autenticação e privacidade para a comunicação de dados.

[2] Padrão, aberto, de criptografia baseado em PGP, utiliza chaves assimétricas.

de montagem, que contém uma espécie de "link"para os arquivos criptografados no diretório fonte, mostrando os arquivos de forma descriptografada (GOUGH, 2003). Tanto o conteúdo dos arquivos quanto seus nomes são criptografados. Há uma chave de volume, que é guardada dentro e fora do diretório fonte, e uma senha é usada para a descriptografar e consequentemente acessar os dados dos arquivos do sistema.

Ele foi projetado por Valient Gough e é de código aberto, tendo contribuições de diversas pessoas. Foi modelado em base do CFS (BLAZE, 1993) (o sistema de arquivos criptográfico original), tendo muitas características provindas deste (LINUXARIA, 2011). Possui dois modos de criptografia, o por *stream* e por blocos. O primeiro é usado para os nomes dos arquivos e blocos parciais nos finais dos arquivos. O segundo é usado para blocos de tamanho fixo, em todo o arquivo, e o tamanho deste bloco pode ser configurado durante a criação do sistema (GOUGH, 2003).

4 Comparação

A fim de realizarmos uma comparação de desempenho entre os dois sistemas estudados, utilizamos o *Iozone* (NORCOTT; CAPPS, 2003), um *benchmark*[3] de sistemas de arquivos que executa diversas rotinas para determinar o *throughput* do sistema testado. O repositório do *EncFS* também fornece um teste semelhante ao executado através do *Iozone*, mas optamos por utilizar uma ferramenta a parte dos sistemas envolvidos para melhor garantia de resultados. Os nossos testes basearam-se nas execuções de leitura e escrita de dados, não sendo testados os demais tipos fornecidas pela ferramenta.

O ambiente onde realizaram-se os testes possui as seguintes configurações:

- Sistema operacional Ubuntu 16.04;
- Processador Intel Core i5 4 núcleos 2.90GHz Cache 6MB;
- Memória RAM de 4GB DDR3;

Criamos o ponto de montagem `/srv` com o *eCryptFS* e o ponto `/encrypted` com o *EncFS*. O *eCryptFS* permite escolher qual algoritmo de criptografia queremos usar e o tamanho dos blocos de forma simples. O *EncFS* também permite configurações personalizadas, como a escolha do algoritmo de criptografia, mas deve-se possuir um conhecimento muito mais avançado pois são necessárias configurações extras. Ambos os sistemas foram configurados para utilizarem o algoritmo AES para criptografia dos arquivos e de seus respectivos nomes, a chave foi configurada com 256 bits e o bloco com 1024 bytes.

Executou-se o seguinte comando para a obtenção dos dados de *throughput*

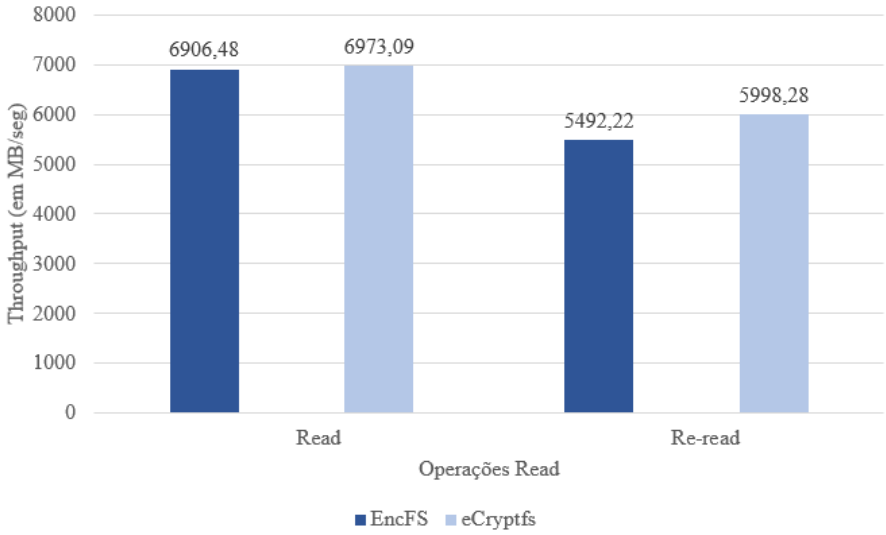
```
# iozone -t 1 -i 0 -i 1 -s 1g -r 128k -F ./mountpoint -S 6000 -b output.xls
```

onde -t configura a quantidade de threads que serão executadas, -i define quais operações serão realizadas (0 - read/re-read, 1 - write/re-write), -s dá o tamanho dos arquivos e -r o tamanho de requisição (leitura e escrita), -F define o caminho do ponto de montagem do sistema a ser testado e -S define o tamanho da memória cache. Por fim, a opção -b faz com que os dados gerados sejam salvos em um arquivo no formato excel. Este comando foi executado por duas vezes para cada sistema de arquivos. Para esta análise

[3] Benchmark é o ato de executar um programa para melhor avaliar o desempenho relativo de um objeto, normalmente executando uma série de testes padrões e ensaios nele.

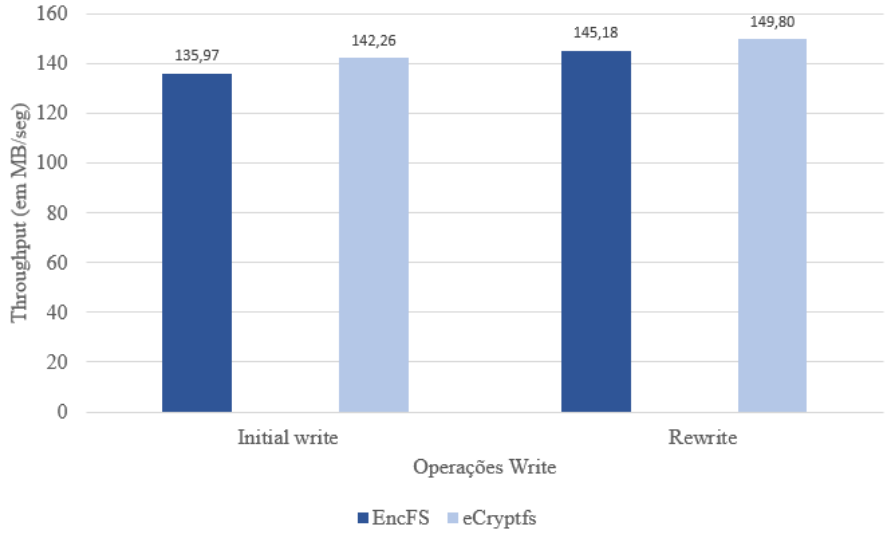
foram feitas as médias dos resultados obtidos em Kbytes/sec e posteriormente convertidos para Mbytes/sec para uma melhor visualização e comparação.

Figura 2 – Taxa de *throughput* nas operações de leitura.



Fonte: os autores.

Figura 3 – Taxa de *throughput* nas operações de escrita.



Fonte: os autores.

A figura 2 nos mostra as taxas de *throughput* médias de operações *read* e *re-read* realizadas em ambos os *file systems*. De acordo com o mesmo, é possível perceber uma mínima vantagem do *eCryptFS* perante o *EncFS* em operações *read*, sendo ela de aproximadamente 1%. Já operações *re-read* o mesmo apresenta uma vantagem um pouco maior, de aproximadamente 8,4%. Em operações *write* e *re-write*, que tem suas taxas mostradas na figura 3, as diferenças apresentadas também não foram muito altas. O *eCryptFS* se mostra 4,4% mais eficiente que o *EncFS* em operações *write* e 3% em

operações *re-write*.

Pelo fato de *eCryptfs* estar alocado em espaço de núcleo, não existe troca de contexto como no *EncFS*, fazendo-o ligeiramente mais rápido em termos de tempo de resposta. Ou seja, a perda de tempo é menor, precisando somente de um tempo extra (se comparado a um *file system* normal) para encontrar a chave a fazer a descriptografia. Todavia, o *EncFS* guarda as informações da criptografia em arquivo central, e não nos *headers* de cada arquivo criptografado como o *eCryptFS* faz, e isso também traz benefícios de performance. Como as diferenças de performance apresentadas não foram de números exorbitantes, estes não desmerecem completamente a eficiência do *EncFS*, pois dependendo do tipo da aplicação essas diferenças são toleráveis se comparadas aos outros ganhos que podem haver.

Além de seu desempenho, existem outros fatores que podem ser utilizados para a escolha de implantação de um ou outro sistema de arquivos criptografado.

4.1 Vantagens e Desvantagens

O *EncFS* é, entre as duas opções apresentadas neste trabalho, a forma mais simples de criptografar arquivos no linux sem necessidade de privilégios especiais (por ser *FUSE-based*) e sob existentes sistemas de arquivos sem precisar realizar modificações. Sua principal vantagem em comparação ao *eCryptFS* é sua portabilidade, somado ao fato de poder criptografar arquivos e armazená-los em ambientes de nuvem (ARCHLINUX, 2014).

Existem algumas aplicações que oferecem uma interface amigável para realizar a montagem do sistema com *EncFS*, e também para que a montagem aconteça já no login do usuário no sistema operacional. Ele é um EFS escalável, pois não possui "volumes"de tamanho fixo – o diretório criptografado cresce conforme mais arquivos são adicionados à ele (ARCHLINUX, 2014).

A principal desvantagem do *EncFS* é que, no momento que um usuário malicioso obter acesso a uma cópia do texto de criptografia, ele possui acesso a todo o sistema (HORNBY, 2014). O atacante pode ver a estrutura do diretório, o número de arquivos contidos nele e seus respectivos tamanhos e quando foram modificados. Todavia, não possui acesso ao dados contidos nos arquivos (LINUXARIA, 2011).

Algumas das vantagens do *eCryptFS*, é o poder de ser um sistema de criptografia que já vem com o kernel, que consegue ter arquivos criptografados individualmente, e utiliza, em sua segurança, *Salting*, que é um dado aleatório (sal) usado junto com a senha, resultando em um *hash*, protege, de uma melhor forma, contra ataques dicionários e pré-computados como o *rainbow table* (ARCHLINUX, 2017a).

Uma desvantagem do *eCryptFS*, é o fato de ao ser usado em *Sparse file* poder gerar arquivos muito maiores, do que se fosse utilizado um outro Sistema de Arquivos Criptografados, a dica é, ou deixar esses arquivos em um diretório não criptografado pelo *eCryptFS*, ou utilizar outro sistema de criptografia (ARCHLINUX, 2017b).

5 Conclusão

A criptografia de arquivos acaba sendo uma ótima alternativa para manter em segurança os dados em uma organização. Dentro das ferramentas descritas e comparadas neste presente trabalho, cada necessidade pode se apresentar mais compatível com uma ou outra opção.

Para aqueles que estão realmente em dúvida entre qual opção escolher, e não possuem necessidades muito específicas que somente um ou outro atenda, o uso do *eCryptFS* é o mais recomendado. Mesmo sendo pequena, a vantagem de desempenho perante ao *EncFS* existe. Além do mais a tecnologia de segurança empregada é superior se comparada ao seu oponente de *user space*. Todavia, para aqueles que precisam criptografar arquivos que serão armazenados em nuvem, o *EncFS* se torna a melhor opção, por sua simplicidade e existências de aplicações *user friendly* para uso e configuração.

Referências

ARCHLINUX. *EncFS - ArchWiki*. 2014. Disponível em: <https://wiki.archlinux.org/index.php/EncFS>. Acesso em: 05 jul. 2017. Citado 2 vezes nas páginas 3 e 6.

ARCHLINUX. *Disk encryption*. 2017. Disponível em: <https://wiki.archlinux.org/index.php/Disk_encryption#Comparison_table>. Acesso em: 6 jul. 2017. Citado na página 6.

ARCHLINUX. *eCryptfs: a Stacked Cryptographic Filesystem*. 2017. Disponível em: <https://wiki.archlinux.org/index.php/ECryptfs>. Acesso em: 6 jul. 2017. Citado na página 6.

BLAZE, M. A cryptographic file system for unix. In: ACM. *Proceedings of the 1st ACM conference on Computer and communications security*. [S.l.], 1993. p. 9–16. Citado na página 4.

GOUGH, V. *EncFS: an Encrypted Filesystem for FUSE*. 2003. Disponível em: <https://vgough.github.io/encfs/>. Acesso em: 05 jul. 2017. Citado 2 vezes nas páginas 2 e 4.

HALCROW, M. *eCryptfs: a Stacked Cryptographic Filesystem*. 2007. Disponível em: <http://www.linuxjournal.com/article/9400>. Acesso em: 30 jun. 2017. Citado na página 2.

HALCROW, M. A. ecryptfs: An enterprise-class encrypted filesystem for linux. In: *Proceedings of the 2005 Linux Symposium*. [S.l.: s.n.], 2005. v. 1, p. 201–218. Citado na página 2.

HORNBY, T. *EncFS Security Audit*. 2014. Disponível em: <https://defuse.ca/audits/encfs.htm>. Acesso em: 05 jul. 2017. Citado na página 6.

KERNEL.ORG. *Documentation of filesystems FUSE*. 2017. Disponível em: <https://www.kernel.org/doc/Documentation/filesystems/fuse.txt>. Acesso em: 30 jun. 2017. Citado na página 3.

LIBFUSE. *libfuse/libfuse: The reference implementation of the Linux FUSE (Filesystem in Userspace) interface*. 2016. Disponível em: <https://github.com/libfuse/libfuse>. Acesso em: 26 jun. 2017. Citado 2 vezes nas páginas 2 e 3.

LINUXARIA. *Introduction to EncFS, Encrypted Filesystem*. 2011. Disponível em: <https://linuxaria.com/article/introduction-to-encfs-encrypted-filesystem>. Acesso em: 05 jul. 2017. Citado 2 vezes nas páginas 4 e 6.

NORCOTT, W. D.; CAPPS, D. *Iozone filesystem benchmark*. 2003. Disponível em: <http://www.iozone.org/>. Acesso em: 05 jul. 2017. Citado na página 4.

RAWAT, U.; KUMAR, S. Distributed encrypting file system for linux in user-space. *International Journal of Computer Network and Information Security*, Modern Education and Computer Science Press, v. 4, n. 8, p. 33, 2012. Citado na página 3.

SYMANTEC. *Cryptographic Filesystems, Part One: Design and Implementation*. 2003. Disponível em: <https://www.symantec.com/connect/articles/cryptographic-filesystems-part-one-design-and-implementation>. Acesso em: 26 jun. 2017. Citado na página 2.

ZADOK, E.; BADULESCU, I.; SHENDER, A. *Cryptfs: A stackable vnode level encryption file system*. [S.l.], 1998. Citado na página 2.