

Encrypted Filesystem Benchmark

6 agosto 2007

Master in Tecnologia del Software Libero e Open Source
http://www.almaweb.unibo.it/os_presentazione.html

Sicurezza Informatica

Anno Accademico 2005/2006

Autori: Roberto A. Foglietta <me@roberto.foglietta.name>,
Pietro Marrone <pietromarrone@gmail.com>

Sommario

Nel provare a completare questa frase "*I nostri dati ...*" potremmo giungere a diverse conclusioni quali ad esempio il fatto che la raccolta sempre più massiccia di dati che ci riguardano è evidentemente un processo sempre più fuori dal nostro controllo sebbene la legislazione stia intervenendo in questo senso.

Ma se la legge protegge la nostra privacy ognuno di noi come tutela la propria? La chiavetta USB che alcuni portano appesa al collo come fosse una collana quanti e quali dati conserva? Siamo certi che in essa non vi sono, se pur lecitamente contenute, anche informazioni sensibili che ci riguardano? Il nostro commercialista, il nostro medico di fiducia o il nostro partner non hanno forse un portatile o un qualsiasi supporto di memorizzazione trasportabile che facilmente potrebbe essergli sottratto, rubato o semplicemente smarrito rivelando di noi qualcosa che vorremmo non fosse assolutamente saputo?

Un uso adeguato della crittografia ci permetterebbe di poter completare la frase così: "*I nostri dati nel caso fossero smarriti sarebbero comunque illeggibili*". Negli ultimi 15 anni molte persone hanno studiato e implementato soluzioni per trasformare la crittografia, quasi fosse stata una forma di magia disponibile solo agli agenti segreti, in uno strumento di massa capace di garantirci quella privacy che gli avanzamenti tecnologici in altri settori, quali telecomunicazioni e supporti di memorizzazione di massa, rischiano di mettere seriamente in crisi.

Indice

1	Ragioni dello studio	4
1.1	Fruibilità e prestazioni	4
1.2	Grado di sicurezza	4
1.3	Altre misure di sicurezza	4
2	La catena della sicurezza	5
2.1	Semplicità e rispetto degli standard	5
3	Lista dei filesystem cifrati	6
3.0.1	Categorizzazione	6
3.1	Destinazione d'uso	6
3.2	Filesystem cifrati impilabili su altri	6
3.3	Filesystem cifrati per interi dispositivi	8
3.4	Filesystem cifrati che usano device strutturati	10
4	Misura delle prestazioni	11
4.1	Metodo	12
4.1.1	Selezione dei filesystem	12
4.1.2	Utilizzo della partizione di SWAP	13
4.1.3	Configurazione e comandi utilizzati	13
4.2	Prestazioni su disco rigido	14
4.3	Prestazioni su chiavetta USB	15
4.3.1	Filesystem di appoggio EXT2 vs EXT3	16
5	Conclusioni	17
5.1	Utilizzo su portatile o fisso	18
5.2	Utilizzo per dispositivi removibili	18

1 Ragioni dello studio

L'uso sempre più estensivo di computer portatili e memorie trasportabili come penne o hard-disk USB impone di considerare nuove misure di sicurezza che vanno oltre alla difesa delle informazioni da attacchi provenienti dalla rete.

Infatti si immagina il danno rispetto alla divulgazione di informazioni sensibili ad esempio nel caso che sia il portatile di un medico a essere sottratto: i dati personali di pazienti affetti da malattie croniche e/o genetiche, a trasmissione sessuale o particolarmente incidenti sotto il profilo sociale potrebbero finire divulgate o finanche a essere usate come strumento di ricatto.

1.1 Fruibilità e prestazioni

Non solo nello scenario particolarmente inquietante appena presentato ma anche in altri più quotidiani come garantire la riservatezza di estratti conto e foto di familiari suggeriscono l'uso di filesystem cifrati che impediscano l'accesso ai dati senza il possesso o la conoscenza di una password. Sebbene sia plausibile che non tutti i filesystem cifrati siano a prova di indagine forense o attacco altamente specializzato è abbastanza ragionevole supporre che, salvo cattive implementazioni, questi strumenti software garantiscano una privacy sufficiente nella maggior parte degli rischi dovuti allo smarrimento, furto o accesso temporaneo non sorvegliato.

Se dal punto di vista della difesa della propria privacy l'uso di un filesystem cifrato su di un dispositivo portatile sia una misura di sicurezza da considerare seriamente non si può nemmeno trascurare l'impatto sulle prestazioni del sistema e sulle abitudini d'uso. Occorre notare che uno strumento di sicurezza affinché sia utilizzato in modo sicuro e appropriato occorre che sia quanto più trasparente e semplice possibile affinché l'utilizzatore lo percepisca come un piccolo o nessun fastidio rispetto al cambiamento delle sue abitudini d'uso in rapporto al vantaggio di una maggiore sicurezza.

1.2 Grado di sicurezza

Tutti i software sono imperfetti e quindi tutti i sistemi sono vulnerabili ma il costo di penetrare un sistema potrebbe essere maggiore del valore in esso contenuto.

Questa considerazione permette di valutare l'uso anche di sistemi di cifratura che al giorno d'oggi sono considerati vulnerabili ma pur sempre sufficientemente forti da convincere colui che è venuto in possesso delle informazioni cifrate che sia più conveniente cancellarle, ad esempio per riutilizzare il portatile o il supporto, piuttosto che cercarle.

Ovviamente tutti coloro che fanno uso di sistemi informatici soggetti al rischio di sottrazione, copie di backup comprese, sui quali mantengono dati altamente confidenziali o di valore economico elevato dovranno occuparsi di fare un'analisi del rischio personalizzata e scegliere la soluzione a loro più congeniale.

1.3 Altre misure di sicurezza

Vale la pena di notare che l'uso di un filesystem cifrato non impedisce la diffusione di virus e non esclude la divulgazione delle informazioni a seguito di un attacco informatico. Infatti il più semplice dei filesystem cifrati richiede una parola chiave per essere leggibile ma una volta immessa qualsiasi applicativo e qualsiasi utente del sistema è in grado di accedere e/o modificare le informazioni in esso contenute. Ovviamente alcuni filesystem adeguatamente configurati su sistemi operativi orientati alla sicurezza permettono di definire una granularità dei permessi assai maggiore per cui effettivamente questa misura di sicurezza si aggiunge alle

altre nel processo di irrobustimento della difesa di un sistema informatico. Nel proseguo del documento si valuterà l'uso della crittografia per mantenere riservati i dati, rispetto ad attacchi non specificatamente mirati, di un singolo utente non esperto che operi su sistemi informatici sotto il suo diretto controllo.

2 La catena della sicurezza

Quando si parla di sicurezza è fondamentale disporre del codice sorgente e della possibilità di verificare che da quel sorgente venga prodotto proprio l'eseguibile in uso. Questo implica che anche del compilatore stesso sia disponibile il sorgente. Inoltre per garantirsi che l'eseguibile faccia esattamente quanto richiesto anche dell'intero sistema operativo deve essere disponibile il sorgente. In questa catena di sicurezza gli anelli estremi sono l'utente da una parte e l'hardware dall'altra.

Per quanto riguarda l'hardware il discorso trascende dagli obbiettivi di questo documento ma si potrebbero fare considerazioni analoghe al software se non fosse che l'hardware ha quella tipica componente di materialità per cui sebbene ad oggi esistono processori a schema aperto, come il SUN OpenSparc T1 [1], è anche vero che nessuno di essi è ancora disponibile per il mercato di massa. Quindi su quest'ultimo punto ci si fida ciecamente del produttore motivo per il quale dal momento in cui una parte della produzione di PC IBM è passata sotto il controllo di un'azienda cinese essi sono stati rimarchiati Lenovo e il governo degli Stati Uniti ha deciso di bandire la loro adozione da parte dell'amministrazione pubblica.

Per quanto riguarda l'utente, limitatamente ai filesystem cifrati, prima di ogni cosa, vale la considerazione che la sicurezza è un processo e non un prodotto per cui la volontà e l'impegno dell'utente affinché i suoi dati siano ragionevolmente protetti sono elementi essenziali. Però questo significa che la facilità di utilizzo di uno strumento di sicurezza è anche un valore molto importante nell'ottica di prevenire errori o comportamenti che potrebbero compromettere seriamente la sicurezza o la persistenza dei dati salvati. Un aspetto che non si dovrebbe trascurare è che la diminuzione del rischio di violazione della riservatezza dei dati aumenta il rischio che essi siano interamente perduti a causa della perdita della parola chiave o a seguito dell'intrinseca difficoltà di un processo di ripristino a fronte di corruzione di anche solo alcuni settori del dispositivo di memorizzazione.

2.1 Semplicità e rispetto degli standard

Sempre in contesto di sicurezza anche la semplicità è un valore in quanto qualsiasi implementazione complessa ha intrinsecamente una maggiore probabilità di contenere errori di programmazione sfruttabili da un attaccante, backdoor e vulnerabilità anche intrinsecamente riferibili all'architettura stessa in quanto la comprensione dell'intero sistema è limitata a pochi.

Per ridurre la complessità di un progetto la modularità unita all'uso di librerie e protocolli standard sono scelte architetture importanti che permettono non solo la creazione di un prodotto sicuro ma di mantenerlo sicuro nel tempo. Nel contesto della sicurezza se congiuntamente alla semplicità si scelgono implementazioni aperte si riducono sensibilmente i rischi esposti al paragrafo precedente perché molti occhi che guardano il codice diminuiscono la probabilità che sia un errore sia un insieme di istruzioni maligne sopravvivano a lungo nel codice.

3 Lista dei filesystem cifrati

In questa sezione viene presentata una lista non esaustiva di alcuni filesystem fra i più noti attualmente rintracciabili in rete, ovviamente a codice sorgente aperto e salvo un'eccezione coperti da licenze libere. Un'altra lista non esaustiva è possibile trovarla su Wikipedia all'indirizzo:

http://en.wikipedia.org/wiki/List_of_cryptographic_file_systems

3.0.1 Categorizzazione

Vi sono alcune caratteristiche attraverso le quali si possono classificare i filesystem cifrati e altre invece che possono essere considerate funzionalità opzionali. Un processo di categorizzazione è sempre relativo a quali caratteristiche si privilegiano. In questo documento la prima divisione è basata su quelli che richiedono di appoggiarsi su un dispositivo fisico o virtuale¹ e quelli invece che si appoggiano a un filesystem esistente. Altre caratteristiche salienti sono la disponibilità su altre piattaforme anche non UNIX-like e la capacità di nascondersi o comunque di offrire la possibilità al proprietario dei dati di negare l'uso di crittografia o dissimularne il reale contenuto crittografato².

3.1 Destinazione d'uso

La destinazione d'uso può privilegiare alcune peculiarità e quindi è doveroso prima d'ogni cosa indicare di quali contesti si è tenuto conto per comporre la lista e in questa lista fare la selezione di quelli di cui verificare le prestazioni.

Si sono esclusi gli scenari di utilizzo in ambiente aziendale e/o in cui la protezione delle informazioni non si riduce solamente all'uso o meno di un filesystem crittografato.

Gli usi invece espressamente considerati sono quelli relativi alla cifratura di partizioni o porzioni di filesystem esistenti in un portatile e la cifratura di memorie di massa rimovibili quali chiavette e hard disk tipicamente collegati mediante bus USB.

Questa scelta implica di proporre la crittografia a un vasto segmento di utenti i quali in maggioranza non sono necessariamente degli esperti. Questo a sua volta implica che la semplicità d'uso e le buone prestazioni sono elementi importanti nel bilancio di sostenibilità della maggiore riservatezza dei dati che essa offre.

3.2 Filesystem cifrati impilabili su altri

L'elenco viene presentato in ordine cronologico di apparizione e contiene anche alcuni filesystem oggi completamente abbandonati ma che in qualche modo era doveroso citare nell'ottica anche di tracciare un percorso evolutivo di questa branca di informatica.

CFS **1993** - Si tratta di un filesystem crittografico di tipo user-level cioè si può utilizzare senza essere necessariamente amministratori della macchina. Le informazioni da proteggere possono essere suddivise in directory ognuna delle quali ha una sua chiave crittografica. CFS si appoggia ad altri filesystem dotati di API POSIX per salvare e accedere alle informazioni dentro a dei file. In particolare è possibile utilizzarlo in combinazione con NFS e con sistemi di backup tradizionali. Per quanto riguarda NFS i dati viaggerebbero criptati in

¹Loop device - http://en.wikipedia.org/wiki/Loop_device

²Plausible deniability - http://en.wikipedia.org/wiki/Plausible_deniability#Use_in_cryptography

rete e sarebbero decodificati solamente sul terminale che ha avanzato la richiesta e che possiede o conosce la chiave. Questo sistema di crittografia quindi è utile anche per conservare informazioni sensibili su server remoti di cui non ci si fida dell'amministrazione o della sicurezza fisica, informatica o degli altri utenti. La criptazione è basata sull'algoritmo DES³ in modalità ECB (Electronic Code Book) combinata insieme a un processo di XOR con una lunga sequenza di numeri pseudo-casuali prodotti e criptati con DES metodo OFB (Output Feedback). Questo in maniera da supplire all'intrinseca debolezza del metodo DES semplice rispetto agli attacchi a forza bruta che oggi giorno è possibile mettere in atto piuttosto agevolmente. Questo filesystem prevede due modalità: "normale" e "alta sicurezza". La seconda protegge in modo più efficiente dai tentativi di rompere la cifratura attraverso il confronto dei blocchi simili ma per come è stata realizzata questa tecnica la modifica incontrollata dei permessi del file criptato porta ad una perdita dell'informazione in esso contenuta, quindi è piuttosto rischiosa e inadatta a persone inesperte. L'autenticazione viene fornita una volta sola per sessione ma se non viene usata scade dopo un certo tempo in maniera da evitare che le informazioni rimangano in chiaro. Dai benchmark presentati in "A Cryptographic File System for Unix" [2] si stima che l'overhead sia del 13% rispetto all'uso del filesystem ospitante. Richiede l'installazione di un demone NFS e si tratta di un filesystem considerato vecchio ma ancora utilizzabile in quei contesti in cui anche una cifratura non particolarmente forte è comunque sufficiente. Può a buon ragione essere considerato il capostipite di quasi tutti i filesystem presentati in questo documento infatti molta della documentazione in quest'ambito fa diretto riferimento ai lavori di Matt Blaze⁴.

- TCFS** **1996** - Inizialmente sviluppato per il kernel 2.0.x all'università di Salerno, successivamente portato al ramo 2.2.x, è da considerarsi ormai obsoleto e al momento della scrittura di questo documento la relativa home page [3] risulta totalmente bianca. Richiede l'installazione di un client NFS.
- CryptFS** **1998** - Versione di filesystem cifrato tipo *proof of concept* ha dato origine al suo successore NCryptfs.
- NCryptFS** **2002** - Anche questo si appoggia su altri filesystem fra i quali ad esempio EXT2, EXT3 o NFS. Permette di usare in modo concorrente diversi algoritmi di cifratura e diversi sistemi di autenticazione capaci anche di gestire gruppi d'utenza. L'uso della cifratura è modulare e l'unico vincolo è che il modulo sia in grado di trasformare un blocco di dati in uno cifrato della medesima lunghezza. La possibilità di scegliere quale implementazione e quale algoritmo di cifratura usare permette di personalizzare il rapporto fra efficienza e sicurezza. L'autenticazione viene fornita una volta sola per sessione ma scade dopo un certo tempo che rimane inutilizzata in maniera da evitare che le informazioni rimangano in chiaro. E' stato implementato utilizzando FiST che è un kit di

³Cifratura DES e varianti - http://www.amagri.it/Crittologia/Crittografia/Algoritmi_crittografici/DES/modalita_funzionamento.htm

⁴Matt Blaze Wikipedia biography - http://en.wikipedia.org/wiki/Matt_Blaze
CFS annuncie - <http://www.crypt0.com/papers/cfs.announce>
CFS tutorial - <http://www.ecn.org/crypto/crypto/tutorial/cfs>
CFS Linux Journal review - <http://www.linuxjournal.com/article/6381>

sviluppo per filesystem impilabili il quale però recentemente pare aver sollevato qualche perplessità⁵ all'interno della comunità degli sviluppatori del kernel. Tale discussione ha avuto un certo impatto anche su altri filesystem⁶. Può essere installato in due modalità fra le quali quella più sicura prevede delle modifiche al kernel quindi non è accessibile all'utenza di base. E' stato sviluppato successivamente a CFS traendone ispirazione in particolare cercando di migliorarne gli aspetti positivi. All'interno del documento in cui viene descritto "NCryptfs: A Secure and Convenient Cryptographic File System" [4] è presentata anche un'interessante tabella di confronto con altri filesystem crittografici dalla quale l'autore conclude che NCryptfs con blowfish abbia un overhead del 5% rispetto all'uso di EXT2. Considerato obsoleto è stato sostituito da eCryptFS.

EncFS

2003 - Filesystem in user space basato sulla libreria FUSE (Filesystem in Userspace) simile concettualmente a CFS ma rispetto a questo ha la possibilità di utilizzare cifrature, considerate oggi giorno, forti come AES e blowfish. L'uso della libreria e modulo FUSE è considerato sicuro anche da chi ha un approccio relativamente paranoico alla privacy e alla segretezza come gli sviluppatori di PhoneBook⁷ che è una parte del più ampio progetto freenet. Supporta gli algoritmi blowfish e AES con chiavi da 128 a 256 bit utilizzando le librerie del progetto OpenSSL⁸ per l'implementazione degli stessi.

eCryptFS

2006 - Basato sulle precedente esperienza di CryptFS, si appoggia su un filesystem esistente per il salvataggio dei dati cifrati. Si presenta come un filesystem di classe *enterprise* anch'esso potrebbe soffrire degli stessi problemi di complessità e manutenzione che sono stati citati per NCryptFS e che eventualmente potrebbero essere imputabili alla comune base rappresentata da FiST. Rispetto ai predecessori fa uso delle API crittografiche offerte dal kernel e si appoggia al lavoro di diversi altri progetti quali Pluggable Authentication Modules (PAM), OpenSSL/GPGME, Trusted Platform Module (TPM) e infine della gestione chiavi di GnuPG in modo del tutto trasparente rispetto all'utente⁹. Si tratta di un progetto ancora in sviluppo attivo perciò richiede l'uso di un kernel Linux particolarmente aggiornato nel quale è stato ufficialmente integrato a partire dalla versione 2.6.19. Non pare al momento della scrittura di questo documento un filesystem cifrato di cui consigliare l'uso agli utenti base ma potrebbe diventarlo in un tempo ragionevole.

3.3 Filesystem cifrati per interi dispositivi

Questi sistemi di cifratura sono indicati principalmente per proteggere dati salvati su memorie removibili come hard disk esterni e chiavette USB ma anche per intere partizioni o insieme al supporto loop device disponibile sotto Linux e altri sistemi operativi *NIX-based. In questa categoria risulta particolarmente interessante la possibilità di condividere i dati cifrati fra sistemi operativi anche molto diversi fra loro.

⁵fonte kerneltrap, vedere i commenti - <http://kerneltrap.org/node/7710>

⁶Diatriba unionfs vs aufs ripresa da Knoppix - <http://knopper.net/knoppix/knoppix51-en.html#aufs>

⁷Freenet PhoneBook manual - <http://www.freenet.org.nz/phonebook/manual.html>

⁸OpenSSL project, home page - <http://www.openssl.org>

⁹Linux Journal, eCryptfs - <http://www.linuxjournal.com/article/9400>

BestCrypt 1993 - si tratta di un software non Open Source realizzata dalla Jetico. E' stato presentato recentemente in un corso fatto all'università di Salerno¹⁰. Pur non avendo analizzato BestCrypt sotto il profilo delle prestazioni ma solo dal confronto della documentazione disponibile si ritiene che TrueCrypt sia una buona alternativa libera. In ogni caso di BestCrypt si parla anche nel Linux Journal già nel giugno 2002 con una recensione¹¹ della versione 1.0b-5, mentre Wikipedia data la sua nascita¹² nel 1993. Si tratta perciò di un prodotto che vede la luce almeno in contemporanea con CFS che può essere considerato il capostipite dei filesystem cifrati.

Pro: supporta la cifratura di interi dispositivi; supporta molti algoritmi di cifratura; crea contenitori leggibili anche da Microsoft Windows se viene utilizzato il filesystem FAT come base sottostante; permette di nascondere il contenitore cifrato.

Contro: non è Open Source quindi occorre fidarsi di un fornitore terzo rispetto agli interessi di sicurezza e privacy di chi ne fa uso; non è compatibile con tutte le piattaforme ma solo per x86; non supporta tutti i filesystem disponibili per Linux ma solo EXT2, EXT3, ReiserFS e Minix; almeno fino alla release 1.5-5 aveva ancora problemi con la serie 2.6 del kernel. Quest'ultima limitazione non lo rende adatto all'uso sotto Linux da utenti base.

TrueCrypt 2000 - è un software Open Source¹³ per utenti Windows e Linux che consente di creare e gestire in modo rapido e sicuro uno o più dischi virtuali crittografati limitando l'accesso alle informazioni solo agli utenti in possesso della password. La sicurezza dell'applicativo deriva dall'utilizzo di algoritmi di cifratura particolarmente resistenti, dal fatto che si utilizzi una passphrase invece di una semplice password ed infine dalla possibilità di nascondere un volume virtuale all'interno di un altro. E' uno strumento pensato per criptare partizioni e dispositivi removibili. Esiste una sua recensione anche su Wikipedia¹⁴.

Pro: è multiplatforma e in particolare permette di accedere allo stesso dispositivo da diversi sistemi operativi; prevede la possibilità di nascondere il proprio utilizzo mediante una tecnica steganografica implicita nel suo sistema di cifratura; l'architettura del sistema rende impossibile determinare la presenza di un volume all'interno di un altro; utilizza diversi sistemi di cifratura più o meno forti o veloci; può essere installato su un dispositivo removibile in modalità di partenza automatica al fine di poter leggere i propri dati anche su computer non predisposti.

Contro: non è in grado di cifrare l'area di swap; quando viene utilizzato su un dispositivo di loop back richiede che la partizione che ospita il contenitore sia formattata con blocchi di 512 byte. Attualmente non è disponibile alcuna versione per i sistemi operativi di casa Apple anche se per MacOSX è in previsione il porting.

¹⁰Università di Salerno, BestCrypt - http://www.dia.unisa.it/~ads/corso-security/www/CORSO-0102/protezione_file_windows/bestcrypt.htm

¹¹Linux Journal recensione BestCrypt - <http://www.linuxjournal.com/article/5938>

¹²BestCrypt su Wikipedia - <http://en.wikipedia.org/wiki/BestCrypt>

¹³TrueCrypt homepage - <http://www.truecrypt.org>

¹⁴TrueCrypt su Wikipedia in inglese - <http://en.wikipedia.org/wiki/TrueCrypt>

3.4 Filesystem cifrati che usano device strutturati

In questa sezione vengono presentati i filesystem cifrati che necessitano dell'uso di dispositivi virtuali loop back o altre astrazioni. Per la loro stessa natura debbono essere integrati nel kernel. Per questo motivo è meno facile individuarne univocamente l'anno di comparsa. In ogni caso tale informazione è relativamente poco importante infatti è noto che il codice inserito nel kernel ufficiale in genere subisce una forte trasformazione del processo di sviluppo sia in termini di velocità sia di organizzazione.

CryptoLoop é stata avanzata la proposta¹⁵ di sostituire Cryptoloop con dm-crypt. Anche se questa decisione probabilmente non sarà attuata nel ramo 2.6 per problemi di retrocompatibilità è comunque opportuno evitarne l'impiego di Cryptoloop in nuove applicazioni in quanto la sua implementazione ha delle vulnerabilità che anche se non facili da violare sono comunque ben note¹⁶, alcune delle quali però non dovrebbero riguardare un utente base^{17 18}.

Pro: supporto nativo all'interno del kernel ramo 2.6; supporta la cifratura di interi dispositivi; supporta molti algoritmi di cifratura; multi piattaforma; supporta tutti i filesystem disponibili per Linux.

Contro: non prevede la possibilità di nascondere l'uso della cifratura; non è compatibile con Microsoft Windows; recentemente è stato scoperto essere vulnerabile ad attacchi ottimizzati con dizionario; Il mount in loop back soffre di alcune limitazioni e banchi.

dm-crypt è un sistema di cifratura dei dispositivi, disponibile dalla versione 2.6 del kernel, che sfrutta l'infrastruttura dei device mapper e le API crittografiche del kernel. Poichè è in grado di cifrare in modo trasparente i device a blocchi fisici supporta la cifratura di interi dischi, partizioni e anche files. Può quindi essere utilizzato in concomitanza con qualsiasi filesystem¹⁹.

Pro: supporta la cifratura di dispositivi interi; multi piattaforma; supporta tutti i file system disponibili per Linux; supporta svariati algoritmi; supporto diretto nel kernel a partire dalla versione 2.6.4; attualmente non risulta vulnerabile ad attacchi ottimizzati con dizionario grazie all'uso di metodi di *key iterations* e *crypto salts*; pur non usando dispositivi in loop back in quanto soffrono di alcuni bug è compatibile con CryptoLoop; l'autenticazione può essere fatta già in fase di partenza del sistema.

Contro: non prevede la possibilità di nascondere l'uso della cifratura; non è compatibile con Microsoft Windows in quanto non ne esiste un porting o una reimplementazione per tali sistemi operativi; sebbene sia stato progettato per resistere agli attacchi sperimentati con un certo successo contro CryptoLoop pare comunque dividerne alcune debolezze²⁰.

¹⁵Kernel Trap, replacing Cryptoloop with dm-crypt - <http://kerneltrap.org/node/2433>

¹⁶LWN, a weak cryptoloop implementation in Linux? - <http://lwn.net/Articles/67216>

¹⁷Wikipedia, Cryptoloop (security) - <http://en.wikipedia.org/wiki/Cryptoloop>

¹⁸Cryptoloop Watermark Exploit - <http://www.securiteam.com/exploits/5UPO1PFPM.html>

¹⁹Wikipedia, dm-crypt - <http://en.wikipedia.org/wiki/Dm-crypt>

²⁰Oopsing cryptoapi on 2.6.* - <http://marc.info/?l=linux-kernel&m=107719798631935&w=2>

LUKS è un acronimo per Linux Unified Key Setup la quale è una specifica per la cifratura dei dischi rigidi creata da Clemens Fruhwirth originariamente concepita per Linux²¹. La specifica LUKS per dm-crypt viene implementata in cryptsetup-luks il quale è da considerarsi un sostituto dell'originale cryptsetup del fornisce tutte le funzionalità originarie e aggiunge tutte le funzionalità offerte da LUKS [5].

Pro: Mentre la maggior parte dei sistemi di cifratura degli hard disk utilizzano formati peculiari, incompatibili e a volte non documentati per il salvataggio dei dati. LUKS si propone di fornire una specifica indipendente dalla piattaforma da adottare quale standard per il salvataggio dati su disco. Però non si tratta solo di una questione di compatibilità ma anche di una corretta gestione sicura e documentata delle password. In questo contesto LUKS è stato progettato in maniera da essere conforme alle specifiche TKS1 [6] che è uno schema di gestione sicura delle chiavi crittografiche addirittura resistente all'analisi forense.

Contro: Si noti però che il documento che descrive le specifiche TKS1 è in stato di bozza e appartiene al medesimo autore di LUKS. Questo da una parte è una cosa positiva in quanto si suppone vi sia completa aderenza fra i due progetti e le due specifiche ma dall'altra parte il fatto che non vi siano altri autori, riferimenti e studi in letteratura circa LUKS e TKS1 getta su questi due strumenti l'ombra del dubbio che invita alla prudenza circa la loro adozione.

Loop AES Loop AES ha un funzionamento identico a CryptoLoop e le differenze principali sono date dalle caratteristiche aggiunte. Questo suggerisce di pensare Loop AES come un'evoluzione sicura e retrocompatibile di CryptoLoop.

Pro: supporta la cifratura di dispositivi interi; multi piattaforma; supporta tutti i file system disponibili per Linux; supporta svariati algoritmi; supporta l'uso di GPG; progettato per non essere vulnerabile ad attacchi ottimizzati con dizionario grazie all'uso di metodi di *key iterations* e *crypto salts*.

Contro: installazione non immediata; non supporta la possibilità di nascondere l'uso della cifratura; non è compatibile con Microsoft Windows; pone un veto alla presenza di alcuni moduli nel kernel in particolare i requisiti richiesti in kernel ed user space rendono impossibile la coesistenza di Loop AES e di CryptoLoop.

4 Misura delle prestazioni

In questa sezione viene presentato il metodo utilizzato per la verifica delle prestazioni dei filesystem cifrati selezionati e le motivazioni della selezione. I test sono stati condotti su due diverse periferiche basate su tecnologie differenti ma entrambe largamente diffuse sul mercato: disco rigido SATA e chiavetta USB da 128 Mb.

²¹LUKS homepage - <http://luks.endorphin.org>

4.1 Metodo

I risultati mostrati sono stati ottenuti compilando su PC IBM dotato di processore Pentium IV a 3 GHz con 512 Mb di RAM e disco rigido SATA Western Digital 40 Gb su Ubuntu²² 7.04 con partizione di swap disattivata il pacchetto BusyBox²³ versione 1.5.0²⁴ per tre volte consecutive mediando i valori riportati con il comando `time`²⁵. In seguito per altre prove sulla versione 1.1 del bus USB verrà usato un PC IBM con Pentium III a 800 MHz e 512 Mb di RAM.

La distribuzione è stata scelta in virtù del fatto che Ubuntu risulta essere la distribuzione più attraente negli ultimi 12 mesi secondo le statistiche²⁶ del sito Distro Watch²⁷ con circa il triplo dei contatti rispetto alla Debian da cui deriva.

Il pacchetto BusyBox è stato scelto in quanto condivide il medesimo sistema di compilazione e configurazione del kernel Linux ma rispetto a quest'ultimo è considerevolmente più piccolo e quindi richiede un tempo decisamente minore per essere compilato.

Il comando `time` nella sua semplicità permette di stabilire, con le dovute cautele mostrate più avanti, il tempo impiegato per il processo di compilazione con la possibilità di distinguere il tempo speso in syscall verso il kernel (system), il tempo speso in applicativo (gcc e ld) e il restante compreso il tempo di attesa I/O il quale è significativamente maggiore nel caso di chiavetta USB (10 - 32%) rispetto al disco rigido (1 - 24%) per tutti i filesystem provati.

4.1.1 Selezione dei filesystem

La selezione dei filesystem è stato un processo progressivo per cui immediatamente si è posto il problema di avere un riferimento univoco e per questo si è scelto EXT2 in quanto è il filesystem nativo di Linux e su questo si sono appoggiati tutti i filesystem cifrati che richiedevano l'utilizzo di un altro filesystem convenzionale. A questa regola si è sottratto TrueCrypt per la semplice ragione che tale filesystem è stato inserito specificatamente per condividere dati cifrati fra sistemi operativi diversi e quindi per TrueCrypt è stato usato il filesystem VFAT a 32bit.

EXT3 è stato aggiunto in quanto filesystem journaled retrocompatibile con EXT2 e quindi rispetto a questo si può considerare quasi come un'opzione. In particolare quest'aggiunta è stata fatta per avere un ulteriore paragone di quanto comportasse in termini di perdita di prestazione attivare un journal piuttosto che la cifratura. Infine è stato lasciato nel quadro dei confronti anche perché stupisce il fatto che sia più veloce di EXT2 se impiegato su una chiavetta USB.

EXT2, EXT3 e VFAT sono stati creati e utilizzati con opzioni di default senza alcuna ottimizzazione manuale. Quando possibile è stato seguito il medesimo approccio anche per i filesystem cifrati con eccezione di eCryptFS per il quale si è voluto valutare l'impatto sulle prestazioni dell'adozione di algoritmi differenti. Questo trattamento speciale nei confronti di eCryptFS è stato concesso in virtù del fatto che pareva essere il più promettente per l'uso su un computer portatile.

Il primo filesystem cifrato inserito nella selezione è stato CFS in quanto esso rappresenta il capostipite e non poteva assolutamente essere trascurato quale termine di paragone con i filesystem sviluppati in seguito. Inoltre esso è disponibile fra i pacchetti installabili da *Ubuntu Universe Repository* quindi facilmente installabile attraverso *synaptic* o *apt*. Quest'ultima moti-

²²Ubuntu home page - <http://www.ubuntu-it.org>

²³BusyBox home page - <http://www.busybox.net>

²⁴BusyBox 1.5.0 - <http://www.busybox.net/downloads/busybox-1.5.0.tar.gz>

²⁵man time - <http://www.die.net/doc/linux/man/man1/time.1.html>

²⁶Rilevazione riferita a "Hit Per Day" del 14 maggio 2007

²⁷Distro Watch home page - <http://distrowatch.com>

vazione è alla base anche dell'inserimento di EncFS, sviluppato 10 anni dopo, il quale per altro non poteva essere trascurato perché facente uso di FUSE²⁸.

eCryptFS è stato inserito nella selezione in quanto recentemente integrato nel kernel Linux e quindi immediatamente disponibile inoltre pareva essere un filesystem che per varietà di algoritmi e per via del ridotto context switching fra user e kernel space potesse rappresentare lo stato dell'arte in questo settore. Purtroppo, invece, da alcuni problemi avuti nei test su chiavetta USB è proprio stato scartato perché valutato non adeguatamente stabile e più avanti saranno spiegate le motivazioni. In ogni caso le misure fatte su disco rigido sono state comunque inserite nella statistica.

Quindi è stato aggiunto TrueCrypt per le motivazioni descritte all'inizio di questa sezione.

Infine, quale rappresentante di quei filesystem basati sull'uso del loop device, anche Crypto-Loop è stato inserito nella selezione preferendolo a Loop AES sebbene sia ritenuto vulnerabile, ma non troppo facilmente, per il fatto che è incluso nel kernel ufficiale.

4.1.2 Utilizzo della partizione di SWAP

Per evitare qualsiasi interferenza è stato deciso di disattivare la partizione di SWAP tout-court ma in generale in un sistema multiutente su cui vi siano informazioni che si vuole mantenere private non ci si può permettere di usare una partizione di SWAP in chiaro. Infatti sebbene solo l'utente *root* possa accedere alla lettura della SWAP è anche vero che nemmeno l'amministratore di sistema dovrebbe poter accedere a dati cifrati degli utenti, almeno in teoria, se non conosce la passphrase. Ovviamente anche `/proc/kcore`²⁹ deve risultare inaccessibile anche per l'amministratore di sistema ma a questo appunto si sconfinava in tutte quelle misure di sicurezza che non sono fra gli obiettivi di questo documento.

4.1.3 Configurazione e comandi utilizzati

A seguire la lista delle impostazioni e dei comandi utilizzati per la verifica delle prestazioni.

busybox il pacchetto è stato inizialmente configurato con `make allyesconfig` e quindi mediante `make menuconfig` sono state rimosse le seguenti selezioni:

- Busybox Settings
 - Build Options
 - * Build BusyBox as a static binary (no shared libs)
 - * Compile all sources at once
 - General Configuration
 - * Support NSA Security Enhanced Linux

time la raccolta dei tempi per essere considerata attendibile deve poter essere ripetibile quindi partire da una condizione nota e terminare con il dispositivo sincronizzato. Le tre prove consecutive verificano sperimentalmente la ripetibilità della misura e il comando seguente realizza le altre condizioni:

```
make clean; sync; time (make; sync)
```

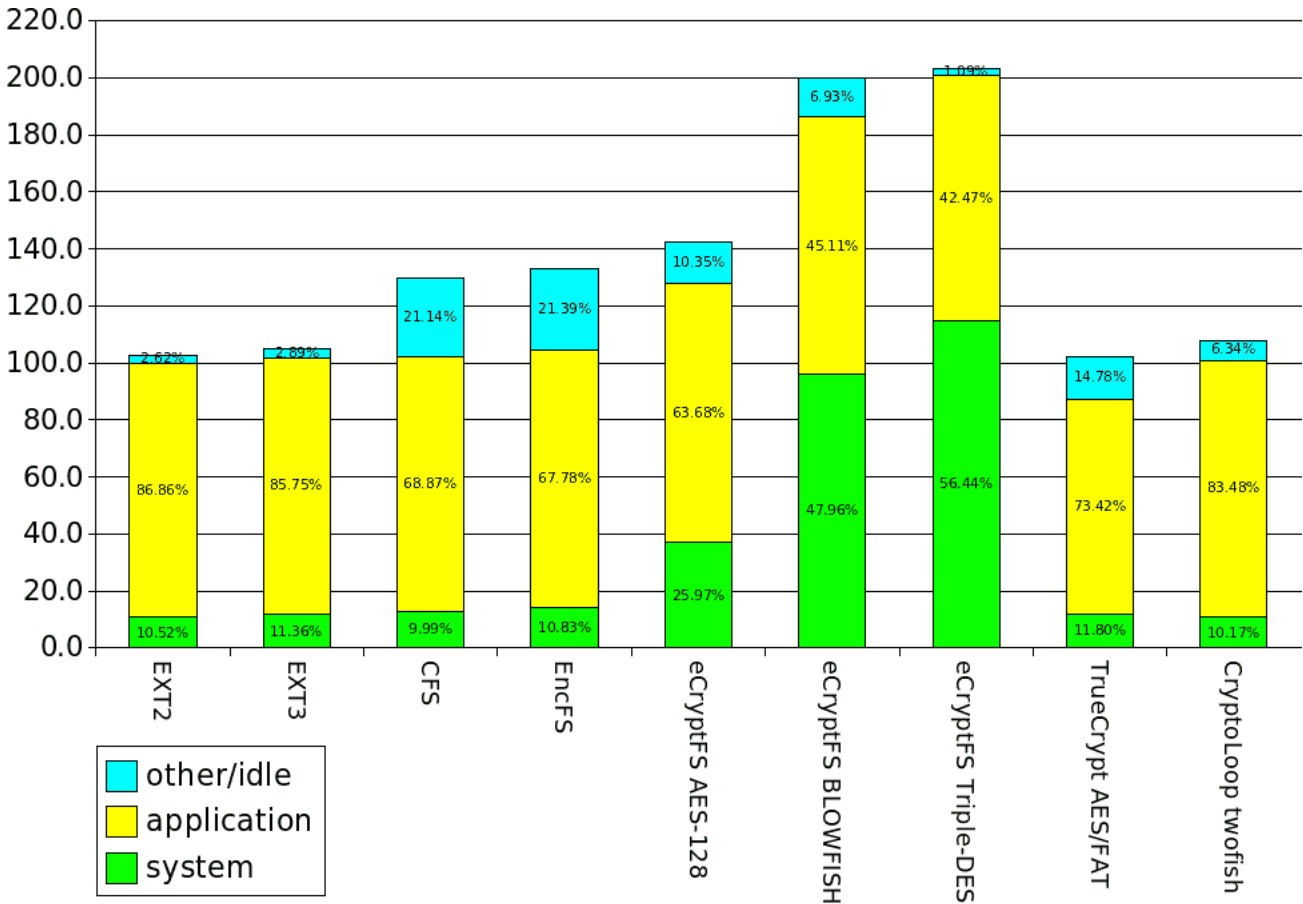
passphrase in tutti i test la passphrase impostata è rimasta uguale: 1234567890abcdef.

²⁸Avendo questo filesystem dimostrato una buona velocità si può estendere il giudizio positivo anche a FUSE.

²⁹SELinux Procfs Analysis - <http://www.nsa.gov/selinux/papers/slinux/node57.html>

4.2 Prestazioni su disco rigido

Per queste prove si è usata un'intera partizione del disco rigido in maniera che in nessun modo la distribuzione dei file preesistenti potesse influire sul test di prestazioni.



Qui accanto viene confrontato il tempo necessario alla compilazione del pacchetto Busy Box su un filesystem cifrato rispetto al tempo necessario per effettuare il medesimo compito sul EXT2 che è stato scelto come il filesystem di riferimento.

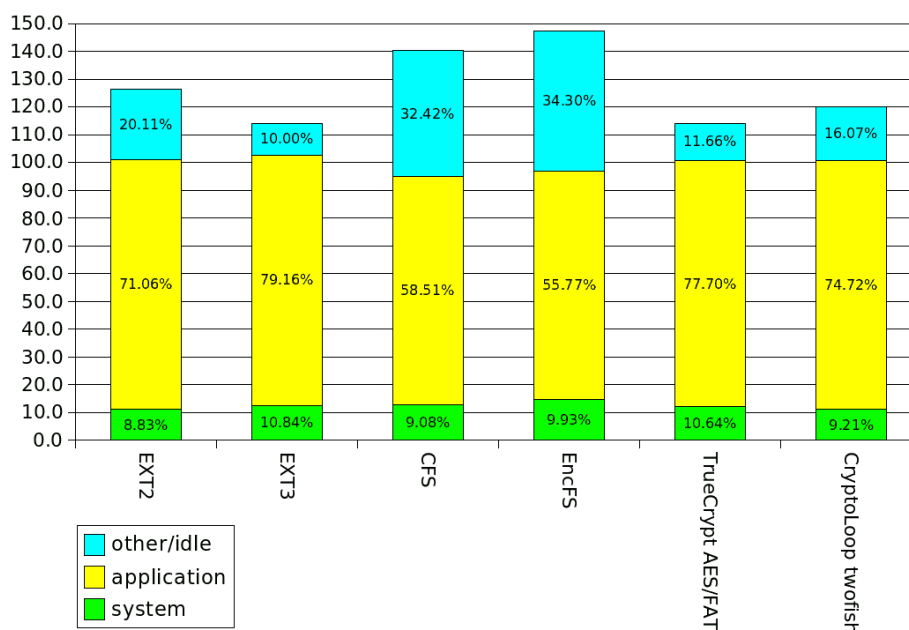
Si nota che l'uso di filesystem impilabili, cioè quelli che si appoggiano all'EXT2 per le operazioni di memorizzazione su disco, implica un richiesta di risorse maggiori fino anche al doppio.

Filesystem su HD	Ratio con EXT2
EXT3	102%
CFS	126%
EncFS	130%
eCrypt AES-128	139%
eCrypt BlowFish	195%
eCrypt Triple-DES	198%
TrueCrypt AES/FAT	100%
CryptLoop TwoFish	105%

Questa però è una caratteristica tipica dovuta all'uso di un hard disk. Più avanti l'uso di supporti USB mostrerà che saranno i tempi di accesso al dispositivo anche legati alla modalità di accesso piuttosto che ai tempi di sistema o di calcolo. I tempi di calcolo anzi si suppone e si verifica rimangono invariati se rimane invariata la potenza di calcolo del processore.

4.3 Prestazioni su chiavetta USB

Per queste prove si è usata l'intero spazio offerto dal dispositivo USB in maniera che in nessun modo la distribuzione dei file preesistenti potesse influire sul test di prestazioni.



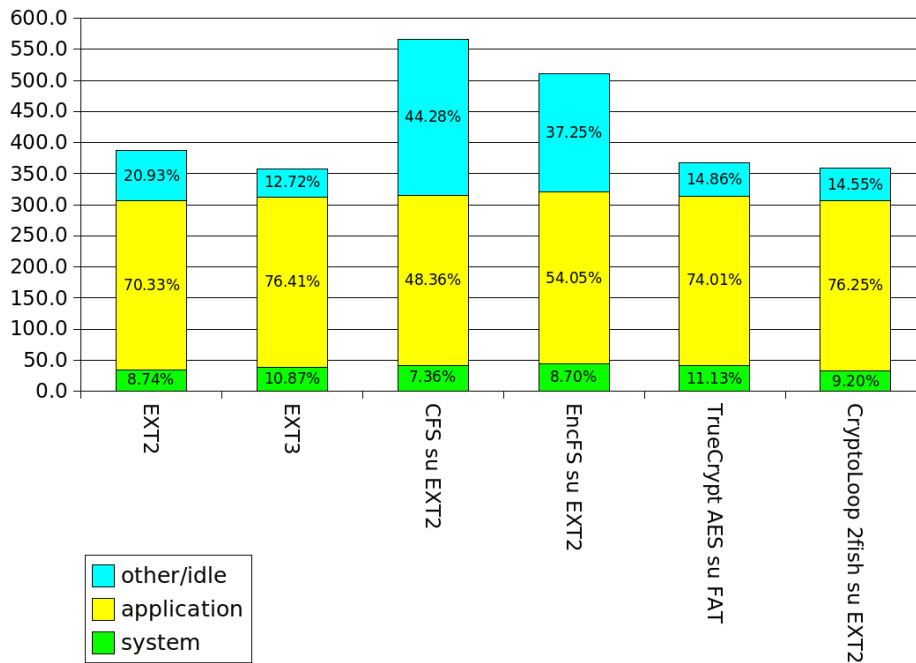
Nella tabella sotto riportata vengono presentati due diversi confronti: ogni filesystem cifrato viene comparato in termini di prestazioni con EXT2 e per ogni tipologia di periferica vengono segnalati i tempi in secondi.

Filesystem	Ratio con EXT2	USB 2.0	HD SATA	USB 1.1	Ratio con EXT2
EXT2	100%	126.6	102.7	387.3	100%
EXT3	90%	114.1	104.9	358.0	92%
CFS	111%	140.4	130.0	566.1	146%
EncFS	116%	147.4	133.1	510.9	132%
TrueCrypt AES/FAT	90%	114.0	102.2	367.9	95%
CryptoLoop TwoFish	95%	120.0	107.8	358.7	93%

Graziosamente il tempo di compilazione con EXT2 su HD SATA risulta essere molto vicino al numero 100 per cui i dati così espressi possono essere letti quasi come fossero anche delle percentuali con un errore di meno del 3%.

Si nota anche che a parità di filesystem EXT2 la compilazione su chiavetta USB 2.0 risulta essere del 23% più lenta rispetto al hard disk che tutto sommato è un termine assai piccolo rispetto al fatto che la banda passante del collegamento SATA è almeno il doppio. Più difficile invece il confronto fra USB 2.0 e 1.1 in quanto è cambiato il processore perciò il tempo di calcolo e di sistema (giallo+verde) si è moltiplicato di quasi esattamente 3 volte (fra 3.01 e 3.04) anche per i filesystem non cifrati. Invece per il tempo che dipende dall'accesso al dispositivo (azzurro) non si trova una relazione proporzionale.

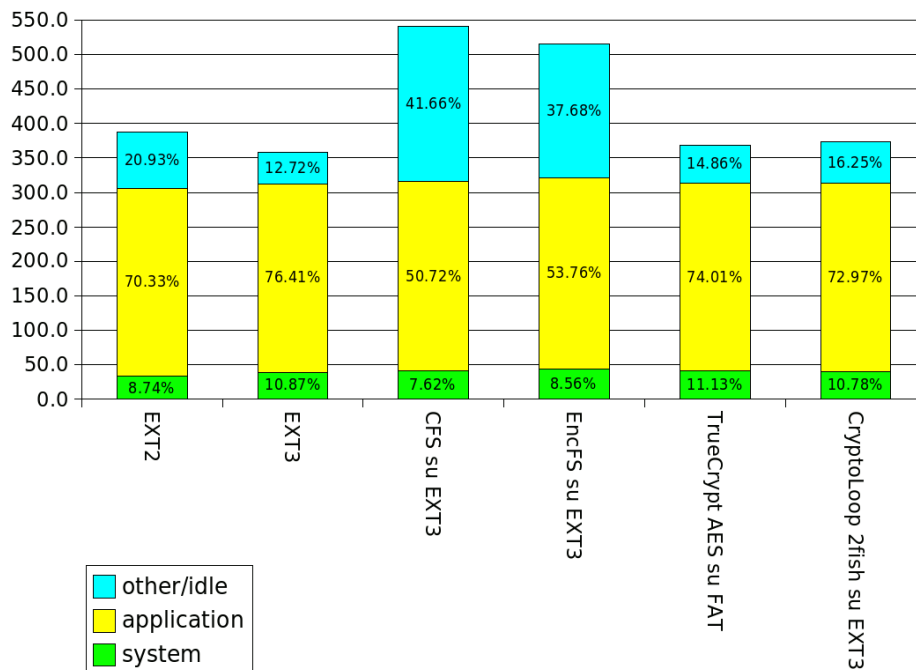
Il rapporto di banda passante fra USB versione 1.1 (12Kbit/s) e la versione 2.0 (480Kbit/s) è 1:40. Questo significa che la modalità di accesso è ancora più significativa e ad esempio EncFS e CFS si sono scambiati il rapporto di convenienza al cambio della versione di USB.



Infatti nei tempi relativi all'I/O l'USB 2.0 varia fra 10% e 34% mentre la 1.1 varia fra 13% e 44% dove le percentuali tengono già conto di una diminuzione della velocità 3x dovuta la processore. Perciò si può affermare che l'USB versione 1.1 è almeno il 30% più lento, a parità di capacità di calcolo del sistema, nell'accesso intensivo (I/O bound stress).

4.3.1 Filesystem di appoggio EXT2 vs EXT3

Si nota che su chiavetta USB 2.0 il filesystem EXT3 appare circa il 10% più performante dell'EXT2 pur essendo il medesimo filesystem con l'aggiunta di un sistema di journaling. Evidentemente la modalità di accesso di EXT3 risulta più adatta al tipo di risposta della chiavetta USB piuttosto che del hard disk.



Il grafico sopra mostra le stesse misure effettuate basandosi su filesystem EXT3 il quale risulta una scelta più performante rispetto a EXT2 in alcuni casi ma non in altri. In generale comunque non pare esserci una differenza rilevante differenza fra i due come mostra la tabella qui sotto riportata:

Filesystem	Ratio con EXT2	su EXT2	Ratio EXT2/EXT3	su EXT3	Ratio con EXT3
EXT2	100%	387.3	100%	387.3	108%
EXT3	92%	358.0	100%	358.0	100%
CFS	146%	566.1	105%	541.5	151%
EncFS	132%	510.9	99%	515.1	144%
TrueCrypt AES/FAT	95%	367.9	100%	367.9	103%
CryptoLoop TwoFish	93%	358.7	96%	373.9	104%

5 Conclusioni

Limitandosi alle prestazioni pare che EXT3 sia più conveniente di EXT2 su dispositivi basati su FLASH e/o collegati via bus USB di circa un 8÷10%. Tale differenza non sembra essere però rilevante quando si vanno a misurare le prestazioni di un eventuale filesystem cifrato che si appoggi a uno di questi due. Infatti in tale circostanza la differenza è del $\pm 5\%$.

In generale un filesystem cifrato può essere anche il 2 volte più lento di un filesystem tradizionale almeno per situazioni d'uso ad intenso accesso. L'uso di un algoritmo piuttosto che un altro è abbastanza significativo, almeno quanto la scelta del filesystem sottostante, ma in generale non ha dimostrato di modificare l'ordine di competitività fra diversi filesystem.

Almeno limitatamente a quelli verificati eCrypt è risultato il più lento e inoltre si è dimostrato particolarmente instabile (almeno con Ubuntu 7.0.4 versione del kernel 2.6.20-15-generic) costringendo l'operatore a effettuare il reboot per smontare la chiavetta USB. Tali instabilità si sono mostrate anche nell'uso di una partizione su disco fisso ma non hanno richiesto il reboot ma solo un intervento da superutente. Per questa ragione è stato scartato nei test su chiavette USB.

Sempre limitatamente a quelli verificati TrueCrypt su FAT ha dimostrato di essere la soluzione più veloce perfomando in modo praticamente identico al filesystem convenzionale EXT3. Inoltre è di quelli verificati l'unico che supporta VFAT come filesystem di appoggio.

Se il sistema di cifratura scelto dipende fortemente dal kernel utilizzato e tale sistema non è ancora stato integrato nel kernel ufficiale si rischia di affrontare una procedura delicata ogni volta che si volesse aggiornare il sistema. Per i portatili non è raro di dover aspettare mesi prima di trovare una configurazione che faccia funzionare egregiamente tutte le periferiche di cui ragionevolmente si sente il bisogno. In questo contesto si tende a non cambiare frequentemente kernel ovviamente rischiando di essere soggetti a falle di sicurezza. Un rischio amplificato dal fatto che un portatile, in quanto tale, finisce per essere collegato a reti diverse e occasionali. In un desktop invece si può aggiungere uno strato di sicurezza proteggendosi attraverso un firewall esterno, magari integrato nel router o nel modem, spostando su di esso l'impegno di tenersi aggiornati e quindi avendo modo di essere meno proattivi nell'aggiornamento del kernel sul desktop.

5.1 Utilizzo su portatile o fisso

Per quanto riguarda l'uso su un portatile o un computer fisso la discriminante effettiva è se si ha una configurazione dual boot e si vuole quindi accedere la partizione cifrata anche da Microsoft Windows. In questo caso l'unica soluzione verificata è TrueCrypt che oltretutto si è dimostrato il migliore anche in termini di performance. Altrimenti la seconda scelta ricade su CryptoLoop.

Varrebbe la pena anche di valutare dm-crypt, eventualmente in previsione di adottare LUKS, in quanto sicuramente sulla carta è una soluzione concettualmente superiore. In questo documento è una soluzione che è stata trascurata sebbene fosse disponibile anche per l'installazione automatica in Ubuntu.

Per questioni di tempo però la valutazione sulla convenienza di adottare dm-crypt, eventualmente insieme a LUKS, oppure Loop AES è lasciata al lettore³⁰.

5.2 Utilizzo per dispositivi removibili

Anche in questo caso la soluzione appare medesima a quella sopra citata e per le stesse ragioni. In un caso reale dove probabilmente i dati cifrati sulla chiavetta USB si volessero condividere fra postazioni diverse, scegliendo TrueCrypt su FAT, è plausibile che il dispositivo di salvataggio estraibile sia partizionato in due in maniera che nella prima partizione siano salvati gli eseguibili necessari all'accesso alla seconda partizione, quella cifrata. Ovviamente se si salvasse su questo dispositivo anche la passphrase l'uso della cifratura sarebbe completamente inutile.

In particolare TrueCrypt possiede anche la modalità di viaggio con auto eseguibilità (autorun) su sistemi Microsoft Windows ma a questo punto occorre fare una considerazione: quale vantaggio si può avere a cifrare i propri dati e poi elaborarli su un sistema operativo insicuro? La meno peggiore delle risposte è che nel caso il sistema in questione sia isolato, cioè non connesso ad alcuna rete, e in uso pressochè esclusivo allora la cifratura proteggerà il contenuto dai rischi di smarrimento del dispositivo stesso.

³⁰Chiunque si sentisse di ripetere i test qui presentati e di includere anche dm-crypt con e senza LUKS è ovviamente invitato a mettersi in contatto con gli autori al fine di integrare il documento con maggiori dati possibili. Qualsiasi altro suggerimento e/o correzione è altrettanto benvenuta. Anche se tale documento non nasce con il proposito di essere regolarmente aggiornato, in tema di sicurezza in particolare, non ci si può esimere dal tenere aggiornare le proprie conoscenze.

Diritto d'autore

Licenza

Questo testo è sottoposto alla licenza Creative Commons Attribuzione 2.5 Italia [9].

Tutti i marchi registrati citati appartengono ai rispettivi proprietari.

La licenza in sintesi

Chiunque è libero di riprodurre, distribuire, comunicare al pubblico, esporre in pubblico, rappresentare, eseguire, recitare e modificare quest'opera alle seguenti condizioni:

- Deve attribuire la paternità dell'opera nei modi indicati dall'autore o da chi gli ha dato l'opera in licenza.
- Ogni volta che usa o distribuisce quest'opera, deve farlo secondo i termini di questa licenza, che va comunicata con chiarezza.
- In ogni caso, è possibile concordare col titolare dei diritti d'autore utilizzi di quest'opera non consentiti da questa licenza.

Permessi

E' permesso all'editore della rivista hackin9 [8] pubblicata dalla Software-Wydawnictwo Sp. z o.o., ul. Bokeserska 1, 02-682 Varsavia, Polonia (in seguito l'Editore) di pubblicare questo documento, anche adattandone l'impaginazione, ma non il contenuto fatta eccezione per l'indice che può essere rimosso insieme ai titoli delle sezioni, a scopo editoriale nella rivista sopracitata. Tale permesso include anche la pubblicazione che dovesse avvenire "a puntate" purchè le varie puntate complessivamente riproducano per intero il testo. All'Editore è permesso di pubblicare un numero illimitato di volte questo lavoro, limitatamente alla rivista sopra citata, senza con ciò trasferire a quest'ultimo nessun diritto esclusivo. Altresì non sono concessi all'Editore altri diritti, salvo previa autorizzazione dei titolari dei diritti d'autore, in particolare non gli è permesso di pubblicare traduzioni riconducibili a questo lavoro e a eventuali adattamenti editoriali di questo lavoro.

Riferimenti bibliografici

- [1] <http://opensparc-t1.sunsource.net> - SUN OpenSparc T1 project
- [2] <http://www.crypto.com/papers/cfs.pdf> - A Cryptographic File System for Unix, Matt Blaze - AT&T Bell Laboratories, 101 Crawfords Corner Road, Room 4G-634, Holmdel, NJ 07733
- [3] <http://tcfs.dia.unisa.it> - Transparent Cryptographic File System, home page
- [4] <http://www.am-utils.org/docs/ncryptfs> - NCryptfs: A Secure and Convenient Cryptographic File System
- [5] <http://luks.endorphin.org/LUKS-on-disk-format.pdf> - LUKS: On-Disk Format Specification, Version 1.1 by Clemens Fruhwirth
- [6] <http://clemens.endorphin.org/TKS1-draft.pdf> - TKS1: An anti-forensic, two level, and iterated key setup scheme by Clemens Fruhwirth
- [7] <http://mareichelt.de/pub/notmine/diskenc.pdf> - Linux for the Information Smuggler by Markku-Juhani O. Saarinen of Helsinki University of Technology
- [8] <http://hakin9.org/it/haking.html> - Hackin9, hard core IT security magazine
- [9] <http://creativecommons.org/licenses/by-nd/2.5/legalcode> - Testo integrale della licenza Creative Commons adottata