TRUECRYPT

FREE OPEN-SOURCE ON-THE-FLY ENCRYPTION

USER'S GUIDE

バージョン情報

TrueCrypt User's Guide, version 3.1a 2005 年 2 月 7 日発行

商標情報

この文書中のすべての登録済みおよび未登録商標は、すべてそれぞれ個々の所有者のものです。

ライセンスと特許情報

TrueCrypt をインストールする、または動作させる(TrueCrypt.exe, TrueCryptSetup.exe, TrueCrypt Format.exe を動作させる)前に、あなたは Licence.txt に書かれているライセンスに同意することが必要です。ライセンスはバイナリとソースコードの配布パッケージにもあります。

CAST5 暗号化アルゴリズムはアメリカ合衆国特許番号 5,511,123[1]に記載されています。しかしながら、世界中で商用、非商用にかかわらずロイヤリティ不要で CAST5 を入手することができます。 [6]。

著作権情報

このソフトウェアの大部分:

A TrueCrypt Foundation Release

詳細情報については、ソースコードに添付された法定の通知を見てください。

制限

TrueCrypt Foundation は、このドキュメントがあなたの要求に合っているとか、情報に誤りがないとかを保証しません。情報は技術的な不正確さがあったり、タイプミスがあったりするかもしれません。

訳者注 :
日本語訳が正確かどうかは保証の限りではありません。ご了承ください。(2005-4-3 : v.3.1a 日本語第 4 版)

目次

まえがき

この文書は、読者がコンピュータのハードウェアとソフトウェアについて一般的知識を持っていると想定しています。通常簡単に理解できるようなことについては、説明不要と判断したところでは説明を省略しています。

はじめに

TrueCrypt は自動即時暗号化するボリューム(データ保存装置)の、作成と維持についてのソフトウェアです。
自動即時暗号化というのは、データが読み出しまたは保存の直前にユーザーの介在なしに自動的に暗号化されるということです。
暗号化されたボリュームのデータは、正しいパスワードまたは暗号化キーがなければ、読むことはできません。
復号されるまでは、TrueCrypt ボリュームは一連の無意味な数値としか見えません。
ファイルシステム全体(すなわち、ファイル名、フォルダー名、すべてのファイルの内容、および空きスペース)が暗号化されます。
TrueCrypt は、復号されたデータをどの記憶装置にも書き込みません。(復号されたデータは臨時に RAM に置かれるだけです)

TrueCrypt ボリューム

二つのタイプの TrueCrypt ボリュームがあります:

- コンテナ
- パーティション/デバイス

TrueCrypt コンテナは、どんな記憶装置にでも存在することができる通常のファイルです。これは内部に、暗号化され完全に独立した仮想ディスク・デバイスを含みます。コンテナはファイル形式のボリュームです。

TrueCrypt パーティションは TrueCrypt で暗号化されたハードディスクのパーティションです。
フロッピーディスク、ZIP ディスク、USB ハードディスク、USB メモリスティック、および他の形式の記憶装置を暗号化することもできます。

新規 TrueCrypt ボリュームの作成

新しく TrueCrypt のファイル形式ボリュームを作ったりパーティションを暗号化(管理者権限が必要)するには、メイン・ウィンドーの「ボリュームの作成」をクリックしてください。TrueCrypt ボリューム作成ウィザードが現れます。
ウィザードは現れたらすぐに、新規ボリュームのためのマスターキー、ソルト、およびIV(initialisation vector 初期化ベクター)やホワイトニング値を作るのに使われる値を生成するためのデータを集めはじめます。収集されたデータは可能なかぎりランダムであるべきで、マウスの動き、マウスボタンのクリック、打鍵などを含み、システムから集められます。(詳細は、「乱数発生機構」を参照)
ウィザードは、新規 TrueCrypt ボリュームを確実に作るために必要な情報とヘルプを提供します。しかしながら、いくつかの項目ではさらに詳細な説明が必要です。

ハッシュ・アルゴリズム

TrueCrypt がどのハッシュ・アルゴリズムを使うかを選択することができます。
選択されたハッシュ・アルゴリズムは(マスターキー、ソルト、IV とホワイトニング値を作る値を生成する)乱数発生機構で使われます。また、ボリュームの新規ヘッダーキーを求めることにも使われます。

TrueCrypt は現在のところ二つのハッシュ・アルゴリズムをサポートしています。; オープン・アカデミック・コミュニティで設計された RIPEMD-160 と、NSA と NIST で設計された SHA-1 です。

ハッシュ関数の出力はけっして直接には暗号化キーとして使われないことに注意してください。詳細は「技術解説」を参照してください。

暗号化アルゴリズム

新規ボリュームを暗号化する暗号化アルゴリズムを選択することができます。詳細は「暗号化アルゴリズム」を参照してください。

クイックフォーマット

「クイックフォーマット」にチェックが入っていない場合、新規ボリュームの各セクターはフォーマットされます。これは、新規ボリュームはランダムなデータで完全に満たされるということを意味します。
クイックフォーマットははるかに速く実行されますが、安全性は劣ります。なぜなら、ボリューム全体がファイルで満たされるまでは、(空き領域がランダムデータで前もって満たされなかった場合には)どれだけのデータがそのボリュームに存在するかがわかってしまうかもしれないからです。クイックフォーマットをしてもよいかどうか判断がつかない場合には、このオプションにチェックをいれないことを勧めます。パーティション/デバイスを暗号化する場合のみ、クイックフォーマットが可能になることに注意してください。
重要: 隠しボリュームを後で作成するつもりのパーティション/デバイスを暗号化する場合は、このオプションにチェックをいれないでください。

クラスタのサイズ

クラスタはファイル配置の単位です。 例えば、1 バイトのファイルのために FAT ファイルシステムで少なくとも 1 個のクラスタを割り当てられます。ファイルがクラスタ境界を越えて大きくなると、別のクラスタが割り当てられます。理論的に、クラスタサイズが大きくなるほど、ディスクにより多く無駄な部分が増えます。(性能はあがりますが) クラスタサイズにどのような値をセットすればいいかわからなければ、初期値のままにしておいてください。

すべてのアルゴリズムを自動でテスト

ボリューム作成ウィザードの暗号化オプションのページにある組み込み済の自己診断機構は TrueCrypt に実装されているすべてのアルゴリズムとすべてのハッシュ・アルゴリズム(HMAC's)を自己診断し、結果を報告します。これらのテストはボリューム生成ウィザードを開始する前に、毎回実行することもできます。
もし何かのエラーがあれば、報告が表示されウィザードはスタートしません。(これはプログラムが破損しているときの新規ボリューム作成を防止するためです)

CD、DVD、および他の書き込み禁止メディアの TrueCrypt ボリューム

TrueCrypt ボリュームを CD、DVD、および他の書き込み禁止メディアに置きたい場合には、まずファイル形式のボリュームをハードディスクに作成してください。それから、CD/DVD 書き込みソフト(Windows XP ならば、OS 標準のシステム・ツールでも可)でそれを CD/DVD に書き込んでください。Windows2000 で読み出し専用メディア(CD/DVD 他)にある TrueCrypt ボリュームをマウントする場合には、TrueCrypt ボリュームを FAT でフォーマットしなくてはならないことを憶えておいてください。(Windows2000 では読み取り専用メディアの NTFS ファイルシステムはマウントできません)

ハードウェア/ソフトウェア・レイドと Windows ダイナミックボリューム

TrueCrypt はハードウェア/ソフトウェア・レイドと同様に Windows のダイナミックボリュームをサポートします。ダイナミックボリュームを TrueCrypt ボリュームとしてフォーマットする場合には(Windows のディスク管理ツールを使って)ダイナミックボリュームを作成したあと、システムの再起動が必要です。そうすれば TrueCrypt ボリューム作成ウィザードの「デバイス選択」に目的のボリュームが表示され、選択できるようになります。

「デバイス選択」ウィンドーで、ダイナミックボリュームは単一のデバイスとしては表示されません。そのかわり、ダイナミックボリュームを構成するすべてのボリュームが表示され、ダイナミックディスク全体をフォーマットするために、どれか一つを選択することができます。

ボリューム作成に関する追加情報

ボリューム作成ウィザードの最終段階で「フォーマット」ボタンをクリックしたあと、システムが追加のランダムデータを得るのに少し間があきます。その後、新規ボリュームのためのマスターキー、ヘッダーキー、ソルト、IV とホワイトニング値を作るのに使われる値などが生成され、マスターキーとヘッダーキーの内容が表示されます。

セキュリティを強化するために、該当のフィールドの右上のチェックボックスにチェックを入れないことで、ランダムプール、マスターキー、ヘッダーキーの内容を表示しないようにできます。

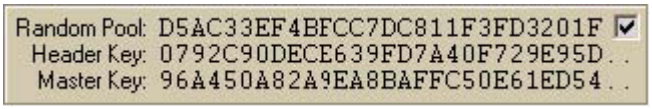

図 1 ボリューム作成ウィザードが表示するランダムプール、マスターキー、ヘッダーの内容

プール/キーの最初の 112 ビットだけが表示されます(全体の内容ではありません)

警告: *ハードドライブパーティション/デバイス全体を暗号化するとき(TrueCrypt ボリュームとしてフォーマットするとき)には、パーティション/デバイスのすべてのデータは失われます。*

重要: 数人のユーザーから、TrueCrypt ボリュームのデータが破損すると報告がありました。その後、これらのユーザーは原因が TrueCrypt ではなくハードウェア(チップセット、USB PCI カード 他)であることを発見しました。ですから、TrueCrypt ボリュームを作ろうとするデバイスに書かれたデータが破損しないか確認することをおすすめします。たとえば、大量のファイル(少なくとも合計 5GB)をコピーし、オリジナルと比較するというようなことです。

TrueCrypt では FAT(FAT12、FAT16、FAT32)か NTFS のボリュームを作成することができます。(しかし、NTFS ボリュームを作成するには、管理者権限が必要です)
TrueCrypt ボリュームは、通常のディスク・デバイスと同じに扱うことができるので、デバイスのアイコンを右クリックしてフォーマットを選択し、いつでも FAT(FAT12、FAT16、FAT32)や NTFS にフォーマットしなおすことができます。

TrueCrypt ボリュームに関する詳細については、「隠しボリューム」も参照してください。

もっともらしい否認

敵対者があなたにパスワードを明かすことを強制するような場合、TrueCrypt は 2 レベルのもっともらしい否認法をあなたに提供します。 TrueCrypt コンテナやパーティションを特定するのは不可能です。暗号化されるまでは TrueCrypt ボリュームはランダムなデータにしか見えません。(TrueCrypt ボリュームであるという「署名」のようなものはありません) ですから、あるファイル、パーティション、デバイスが TrueCrypt ボリュームであり暗号化されているとは証明することはできません。第二レベルの「もっともらしい否認」は隠しボリューム機能によって提供されます。(詳細は「隠しボリューム」を参照)

TrueCrypt コンテナファイルは特定の拡張子を必要としません。どんな拡張子(たとえば raw, .dat, .iso, .img, .rnd, .tc)でもつけることができます。また、拡張子なしでもかまいません。TrueCrypt は拡張子には影響されません。もし「もっともらしい否認」が必要なら、TrueCrypt ボリュームに .tc という拡張子をつけるべきではありません。(この拡張子は TrueCrypt に関連づけられているからです)

ハードディスク・パーティションを TrueCrypt ボリュームとしてフォーマットする場合、パーティション・テーブル(パーティション・タイプを含む)は変更されません。TrueCrypt パーティションを使い「もっともらしい否認」が必要なら、以下の手順にしたがってください。(Windows XP の場合)

1) そのデバイスでパーティション作成が可能なことを確認してください。
   いくつかのリムーバブルメディア(USB メモリスティック他)ではパーティション作成はできません。そのかわり、下記の手順でデバイス全体を暗号化してください。(「デバイス選択」でそれを選んで、あとは通常に進めてください)
2) あなたが管理者権限を持っているか確認してください。
3) デスクトップにあるマイコンピュータのアイコンを右クリックして「管理」を選択するか、スタートから「コンピュータの管理」を選択してください。
4) 左側のリストから「ディスクの管理」(ストレージがツリー表示されています)をクリックしてください。
5) TrueCrypt ボリュームとしてフォーマットしたいパーティションがすでに作成されていれば、右クリックで「パーティション削除」を選択、もしパーティションが未作成ならステップ 4 から続けてください。
6) 未使用領域(未割当と表示)を右クリック、「新規パーティション」を選択。
7) 新規パーティション・ウィザードが表示されるので、その指示にしたがってください。ウィザードの「ドライブレターまたはパスの指定」では、「ドライブ文字またはドライブパスを割り当てない」を選択して、次に進んでください。
8) 「このパーティションをフォーマットしない」を選択し、「次へ」をクリック。
9) 「完了」をクリック。
10) パーティションは「正常」と表示されます。このパーティションはフォーマットされていないので、どんなランダム値でも含むことができます。例としては、最後のパーティション操作以降のデータなどです。ですから、未フォーマット・パーティションと TrueCrypt パーティションに違いを見つけることはできません。これで、パーティションを TrueCrypt ボリュームとしてフォーマットできるようになりました。(フォーマットするには、メインプログラム・ウィンドーの「ボリュームの作成」をクリックし、ボリューム作成ウィザードの指示に従ってください)

    補足: もし未フォーマット・パーティションの代わりに TrueCrypt パーティシンを NTFS か FAT でフォーマットした場合には、そのパーティションは NTFS か FAT の破損パーティションとして見えます。このようなパーティションは未フォーマットのままであるよりも、暗号化されているのではないかと疑われやすいでしょう。(上記参照)

ファイル形式コンテナのタイムスタンプ(コンテナが最後にアクセスまたは変更された日付と時刻)はTrueCryptがコンテナをアクセスすること(アンマウント、マウントの試行、パスワードの変更、隠しボリュームの作成)では更新[*]されません。

[*] Windowsでは、ファイルのプロパティでコンテナの最終アクセス日時を変更できます。

隠しボリューム

誰かが暗号化ボリュームのパスワードを明かすよう強要するかもしれません。それを拒否できない状況、たとえば敵対者が暴力に訴えてパスワードを求めてくるようなこともありえます。そこで、いわゆる「隠しボリューム」を使うことで、ボリュームのパスワードを明かさずに策略で解決する方法があります。

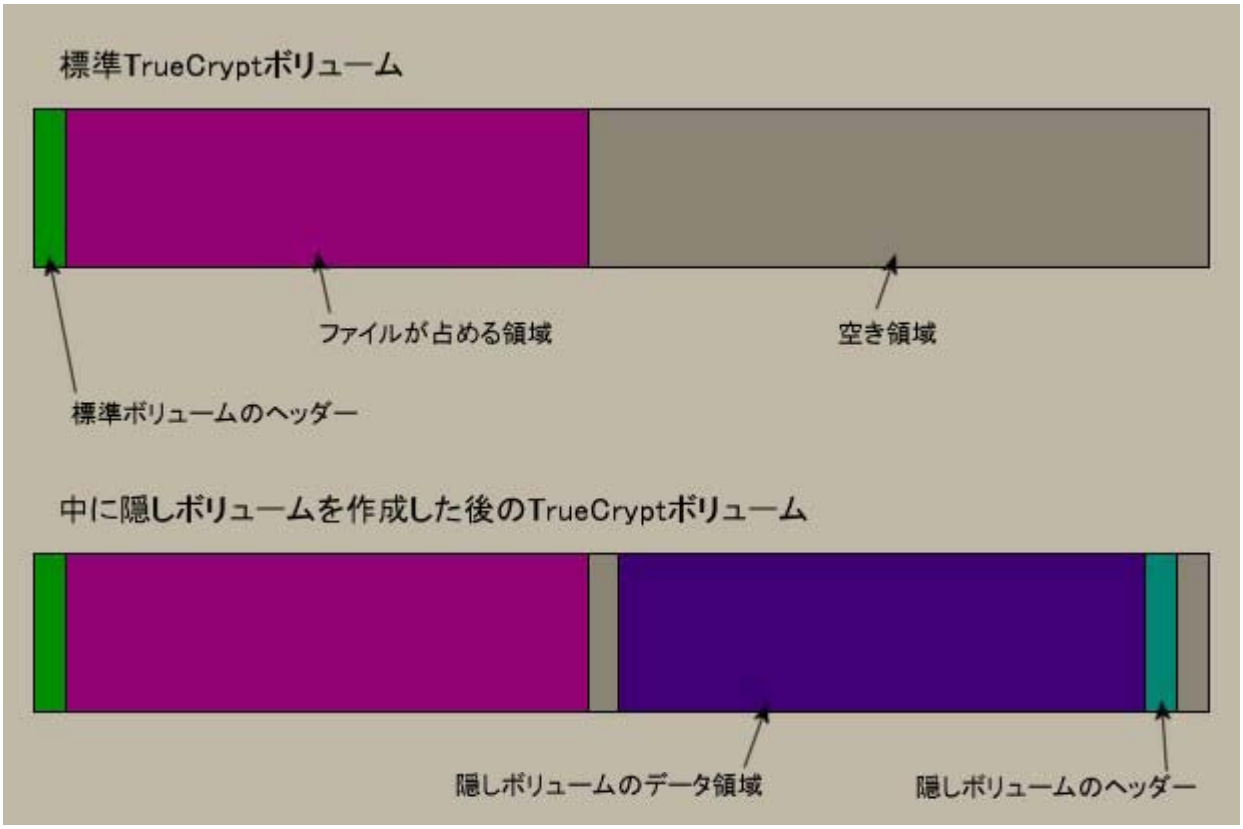

図 2 隠しボリューム作成前後の標準 TrueCrypt ボリュームの状態

他のTrueCryptボリュームの空き領域にTrueCryptボリュームを作るというのが、ポイントです。外殻ボリュームがマウントされるとき、それが隠しボリュームを含むかどうかを判断することはできません。なぜなら、どのTrueCryptボリュームの空き領域も作成時*にランダム値で埋められている(クイックフォーマット時を除く)からです。そして、隠しボリュームのどの部分もランダムデータと区別できません。

* TrueCryptボリュームのフォーマット直前に、臨時の暗号化キー、ある量のプレーンテキスト、IVとホワイトニング・シードが内蔵の乱数発生機構で生成されます。(これらの項目はRAMにのみ置かれ、フォーマット終了後に破棄されます) ユーザーが選択した暗号化アルゴリズムは臨時キーで初期化され、それで暗号化されたテキストがボリュームの空き領域を埋めます。IVはボリュームヘッダーからIVシードが得られない場合以外には通常に生成されます。しかし、このような場合には乱数発生機構によって生成されます。ホワイトニングは通常のように(「ホワイトニング」参照)おこなわれますが、ホワイトニング値は乱数発生機構で生成された値から得られます。

隠しボリュームのパスワードは、外殻ボリュームのパスワードとは異なったものでなくてはいけません。隠しボリュームを作成する前に、外殻ボリュームには本当には隠そうとは思っていない何か秘密情報らしいファイルをいくつかコピーしておいてください。これらのファイルは、あなたにパスワードを明かすことを強要する人に見せるためのものです。隠しボリュームのパスワードは守り、外殻ボリュームのものだけを明かせばいいのです。
本当に秘密にしたいファイルは隠しボリュームに入れてください。

ユーザーは、標準TrueCryptボリュームと同じ手順で隠しボリュームをマウントできます。外殻ボリュームを選択してパスワードを入力してください。隠しボリュームがマウントされるのか、外殻ボリュームがマウントされるのかは、入力されたパスワードで決まります。外殻ボリューム用パスワードが入力されれば外殻ボリュームがマウントされ、隠しボリューム用パスワードが入力されれば隠しボリュームがマウントされます。*

隠しボリュームは TrueCrypt のどの形式のボリュームででも作成できます。ファイル形式でもパーティション/デバイス形式(管理者権限が必要)でもかまいません。TrueCrypt 隠しボリュームを作成するには、メインプログラム・ウィンドーで「ボリューム作成」をクリックし、「隠しボリューム作成」を選択してください。ウィザードは TrueCrypt 隠しボリューム作成に必要なすべての情報やヘルプを提供します。

隠しボリュームが外殻ボリュームのデータを上書きしてしまわないように隠しボリュームの大きさを決めるのは初心者には非常に難しい(特に断片化している場合)ので、ボリューム作成ウィザードは隠しボリューム作成の前に自動で外殻ボリュームのクラスタ配置をスキャンし隠しボリュームの作成可能な最大サイズを決めます。
補足: ウィザードはクラスタ配置をスキャンし、外殻ボリュームと終端が一致する連続した空き領域の大きさを決定します。この領域が隠しボリュームに適用され、この領域の大きさが隠しボリュームの大きさの上限となります。

警告: *隠しボリュームを作る前に外殻ボリュームへ書き込んだファイル以外に、ファイルを外殻ボリュームにコピーしたり新規にファイルを作成したりしないでください。もし、そうしてしまうと隠しボリュームの一部を上書きし破損させてしまいます。*
しかし、隠しボリュームを作成したあとでも、外殻ボリュームのファイルを開いたり、削除、名前の変更は可能です。同じ外部フォルダーの中で、あるフォルダーから他のフォルダーへ移動させることもできます。外殻ボリュームのファイルの内容を変更することは可能ではありますが、*絶対にファイルサイズが変わってはいけない*ということに注意してください。また、エディターなど何かのソフトが臨時ファイルを外殻ボリュームに書かないように注意してください。

隠しボリュームは FAT の TrueCrypt ボリュームにだけ作成できます。(外殻ボリュームのファイルシステムは FAT12, FAT16, または FAT32 に制限されます) NTFS ファイルシステムは(FAT とは対照的に)ボリューム全体に散らばっていろいろなデータを格納するので、隠しボリュームを作る余地を残してくれません。したがって、ボリューム作成ウィザードは外殻ボリュームのファイルシステムとして NTFS を選択することを防止します。隠しボリューム自体はお好みのどのようなファイルシステムでもかわまず、(ファイル形式の)外殻ボリュームはどんなファイルシステムにでも格納できます。
補足: 外殻ボリュームがなぜ FAT なのかと聞かれたら、すべての設定を初期値(デフォルト)のままにしたから、と答えてください。(TrueCrypt ではどのボリュームも初期値は FAT です)

* TrueCryptは入力されたパスワードでまず標準ボリュームのヘッダーを解読しようとします。それに失敗すると、そのパスワードを使って隠しボリュームのヘッダーがあると思われる場所(ボリュームの最後から3番目のセクター)を解読しようとします。それが成功すると外殻ボリュームの中にある隠しボリュームの大きさと位置の情報を得て、隠しボリュームがマウントされます。

警告: もし敵対者が、アンマウントされた TrueCrypt ボリュームを数回にわたって入手したとすると、ボリュームのどのセクターに変更があったかをつきとめることができます。あなたがファイルをつくったり、コピーしたり、ファイルの更新、削除、リネーム、移動などで隠しボリュームの内容に変更を加え相手が新旧のボリューム全体を比較すると、外殻ボリュームのパスワードを教えたにもかかわらずこれらのセクターの内容の変更について追求されるかもしれません。あなたの回答如何によっては、相手はボリュームに隠されたボリュームがあると疑うかもしれません。

警告: *隠しボリュームを作ろうとするパーティション/デバイスを暗号化するときには、クイックフォーマットはしないでください。*

もし隠しボリュームを作るのになにか問題があれば、「問題が起こったら」を参照してください。

補足: ファイル形式コンテナのタイムスタンプ(コンテナが最後にアクセスまたは変更された日付と時刻)は TrueCrypt がコンテナをアクセスすること(アンマウント、マウントの試行、パスワードの変更、隠しボリュームの作成)では更新されません。これは通常ボリュームでも隠しボリュームでも同じです。

主プログラム・ウィンドー

ファイルの選択

ファイル形式の TrueType ボリュームを選びます。選択したあとで、「マウント」(下記参照)をクリックすることでマウントできます。ボリュームのアイコンを TrueCrypt.exe のアイコンにドラッグ&ドロップすれば、TrueCrypt が起動します。TrueCrypt プログラム・ウィンドーにドラッグすることもできます。

デバイスの選択

TrueCrypt パーティションか記憶デバイス(たとえばフロッピーディスクや USB メモリスティック)を選びます。その後、「マウント」(下記参照)をクリックすることでマウントできます。「デバイスの選択」のかわりに、マウントしたいコンテナを TrueCrypt.exe のアイコンにドラッグすることもできます。

補足: TrueCrypt パーティション/デバイスをマウントするもっと簡単な方法があります。「デバイスの自動マウント」を参照してください。

マウント

TrueCrypt ボリュームをマウントするには、メイン・ウィンドーから空いているドライブレターを選んでください。
そして TrueCrypt ボリュームであるファイルかデバイスを選び、「マウント」をクリックします。TrueCrypt はキャッシュにパスワードがあればそれを使ってマウントしようとします。キャッシュになければ、ユーザーにパスワード入力を要求します。正しいパスワードを入力すれば、マウントされることになります。
重要: Windows XP/2000/2003 ではユーザーの切替やログオフでは、正常にマウントされた TrueCrypt ボリュームはアンマウントされないことに注意してください。また、TrueCrypt アプリケーションを終了しても TrueCrypt ドライバーが機能しており、TrueCrypt ボリュームはアンマウントされません。

デバイスの自動マウント

この機能を使うと(「デバイス選択」を使って)目的のパーティション/デバイスを選択しなくても TrueCrypt パーティション/デバイスをマウントすることができます。TrueCrypt はあなたのシステムの有効なパーティション/デバイスを探して、それぞれを TrueCrypt ボリュームとしてマウントしようとします。TrueCrypt パーティション/デバイスであるかどうかは特定できず、使われている暗号の種類も特定できないことに注意してください。ですから、プログラムは目的の TrueCrypt パーティションを直接には見つけることはできません。そのかわり、TrueCrypt は暗号化されていてもいなくても、すべての暗号化アルゴリズムと(存在するなら)キャッシュにあるパスワードを使って、パーティション/デバイスを一つずつ試します。このため遅いマシンでは、このプロセスに長時間かかることは了承してください。ドライブレターはメイン・ウィンドーのドライブリストで選択された最初のものに割り当てられます。入力したパスワードが不正であれば、(存在すれば)キャッシュのパスワードを使ってマウントを試行します。デバイスの自動マウントでは、空のパスワード入力の場合にはキャッシュのパスワードのみが使われることになります。

アンマウント

TrueCrypt ボリュームをアンマウントすると、そのボリュームのデータにはまったくアクセスできなくなります。アンマウントを実行するには、TrueCrypt ボリュームを選んで、「アンマウント」をクリックしてください。

すべてアンマウント

現在マウントされているすべての TrueCrypt ボリュームをアンマウントします。

記憶したパスワードの消去

ドライバーのメモリに記憶されているすべてのパスワードを消去します。キャッシュにパスワードが記憶されていなければ、このボタンは押せないようになっています。直近にマウントに成功した TrueCrypt ボリュームのパスワードを 4 件までキャッシュできます。これにより、繰り返してパスワードを入力しなくてもボリュームのマウントができます。TrueCrypt は絶対にパスワードをディスクに保存しません。パスワード用キャッシュは RAM に置きます。(しかし、「安全上の注意」を参照してください) パスワード・キャッシュはツールメニューの設定で、有効にも無効にもできます。

パスワードの変更

現在選ばれている TrueCrypt ボリュームのパスワードを変更することができます。(通常ボリュームか隠しボリュームかを問いません) ヘッダーキーのみが変更され、マスターキーは変更されません。ですから、再フォーマットは必要なく、実行もされません。(パスワード変更はデータを失うことはなく、ほんの数秒で完了します)

敵対者があなたのパスワードを知ってあなたのボリュームにアクセスすると、マスターキーを入手することが可能になるかもしれません。そうなると、あなたがパスワードを変更しても相手はボリュームを復号することができるようになるかもしれません。 (なぜなら、マスターキーは変更されないから) このような場合は、新しい TrueCrypt ボリュームを作り、すべてのファイルを古いボリュームから新しいほうへ移動してください。

TrueCrypt ボリュームのパスワードを変更するには、「ファイル選択」か「デバイス選択」をクリックし、ボリュームを選択してから「パスワードの変更」をクリックしてください。

PKCS-5 PRF アルゴリズム: ボリュームのパスワードを変更するとき、HMAC ハッシュ・アルゴリズムを選ぶこともできます。これは新しいボリュームのヘッダーキー(詳細は「ヘッダーキーの生成、ソルト、反復回数」を参照)や新しいソルト(詳細は「乱数発生機構」を参照)を作るときに使われるアルゴリズムです。

ボリュームの履歴を保存しない

これがチェックされていると、マウントしたボリュームの直近 8 件のファイル名やパスは履歴に保存されません。(履歴はボリュームのコンボボックスをクリックすると表示されます) しかし、これをチェックしても Windows のファイル選択履歴にファイル名が保存されることを防ぐことはできません。これを避けるためには「ファイルの選択」を使わず、コンテナのアイコンを TrueCrypt アイコンへドラッグしてください。(TrueCrypt は自動的に起動します)

終了

TrueCrypt アプリケーションを終了します。ドライバーは継続して動作し、TrueCrypt ボリュームはアンマウントされません。

プログラムメニュー

注意: 自明のメニュー項目は、このドキュメントでは説明しません。

ファイル -> 終了

TrueCrypt アプリケーションを終了します。ドライバーは継続して動作し、どの TrueCrypt ボリュームもアンマウントされません。トラベラーモードの場合には、TrueCrypt ドライバーは必要がなくなればアンロードされます。(メインアプリケーションとボリューム作成ウィザードが終了し、TrueCrypt ボリュームがまったくマウントされていない場合)

ボリューム => デバイスのボリュームをすべて自動でマウント

*「デバイスの自動マウント」*を参照。

ツール -> ボリュームの履歴を消去

最後にマウントに成功したボリューム 8 件のファイル名やパスの履歴を消去します。

ツール -> トラベラーディスクの作成

*「トラベラーモード」*を参照。

ツール -> 設定

終了時に記憶パスワードを消去する

これが有効になっていると、ドライバーのメモリ中に保持されているパスワードは TrueCrypt 終了時に消去されます。

ドライバーのメモリ内にパスワードを記憶する

これがチェックされていると、直近に正常にマウントした TrueCrypt ボリュームのパスワードを 4 件までドライバーのメモリ中にキャッシュします。

マウント時にボリュームのドライブを開く

このオプションがチェックされていると、TrueCrypt ボリュームが正常にマウントされたあと、エクスプローラのウィンドーが自動的にそのボリュームのルートディレクトリ（たとえば T:¥）を表示します。

アンマウント時にボリュームのウィンドーを閉じる

TrueCrypt ボリュームをアンマウントしたいときに、そのボリュームにある何かのファイルかフォルダーが使用中でロックされているためにアンマウントできないということがあります。エクスプローラウィンドーが TrueCrypt ボリュームにあるディレクトリを表示しているときも同様です。このオプションがチェックされていると、そのようなウィンドーはアンマウント前に自動的にクローズされ、ユーザーが手動でクローズする必要がありません。

TrueCrypt ボリュームのマウント

まだマウントしたことがなければ、「マウント」と「デバイスの自動マウント」を読んでください。

ドライバーのメモリ内にパスワードを記憶する

このオプションはパスワード入力ダイアログでセットされます。チェックが入っていれば、入力されたボリュームパスワードは(正しい場合)ドライバーのメモリに記憶されます。あとでボリュームをマウントするときには、記憶されたパスワードが使われるので、パスワードの再入力は不要となります。4 件までパスワードを記憶することができます。TrueCrypt はどんなパスワードもディスクに保存することはありません。(RAM に臨時保存するだけです) パスワード記憶をオフに切り換えても、記憶そのものを消去するわけではないことに注意してください。(*記憶したパスワードの消去*をクリックすることで消去できます)

マウントオプション

マウントオプションはボリュームのマウントのされかたに影響します。*マウントオプション*ダイアログはパスワード入力ダイアログの *マウントオプション*をクリックすることで開きます。正しいパスワードが記憶されると、*マウント*をクリックすることでボリュームは自動的にマウントされます。記憶されたパスワードを使ってマウントされているボリュームのマウントオプションを変更したい場合には、*コントロール(Ctrl)*を押しながら *マウント*をクリックするか、*ボリュームメニューの オプションを指定して ボリュームをマウント*を選択してください。マウントオプションの初期値は、メインプログラム設定(*ツール -> 設定*)で設定しなおすことができます。以下のマウントオプションの設定が可能です。

ボリュームを読み取り専用でマウント

チェックが入っていると、マウントされたボリュームにはいっさい書き込みができません。なお、Windows 2000 では NTFS ボリュームを読み取り専用ではマウントできません。

ボリュームをリムーバブルメディアとしてマウント

Windows が勝手に *Recycled* や *System Volume Information* といったフォルダー(これらはゴミ箱やシステム回復機能のために作られます)を作ることを防止したいなら、このオプションにチェックを入れてください。

トラベラーモード

TrueCrypt はいわゆるトラベラー(旅行者)モードで動作させることができます。これは、TrueCrypt を稼働する OS に対してインストールしなくていいということです。しかし、次の 2 項目は憶えておいてください。

1) TrueCrypt をトラベラーモードで動かすには管理者権限が必要
2) トラベラーモードで起動したとしても、レジストリファイルを検査すれば、Windows で TrueCrypt を使ったということがわかってしまうかもしれません。

この問題に対処するためには、BartPE を使うことをすすめます。また「*FAQ*」の「*OS の起動パーティションを暗号化できますか?*」を参照してください。

TrueCrypt トラベラーモードを使うには、二つの方法があります。

1) バイナリ配布パッケージを展開し、(インストールせずに)直接 TrueCrypt.exe を走らせる。
2) 特別なトラベラーディスクを作る*トラベラーディスク作成*の機能を使い、そこから TrueCrypt を起動する。

2 番目のほうがいくつか有利な点があり、以下にそれらについて説明します。

ツール –> トラベラーディスク作成

オートラン設定 (autorun.inf)

トラベラーディスクが挿入されると自動的に TrueCrypt を起動したり、自動的に特定の TrueCrypt ボリュームをマウントするように設定できます。これは、トラベラーディスクに *autorun.inf* という特別なスクリプトファイルを作ることで可能になります。このファイルはトラベラーディスクが挿入されるつど OS によって自動実行されます。ただし、これは CD/DVD のようなリムーバブルメディアのみで、それらが読取可能な場合のみに動作します。(Windows XP SP2 では USB メモリスティックでも機能します)
この機能を有効にするためは、*autorun.inf*ファイルはトラベラーディスクのルートディレクトリに置かれなくてはならないことに注意してください。(たとえば、G:¥, X:¥, Y:¥ などです)

ボリューム作成ウィザードを含める

トラベラーディスクから起動した TrueCrypt を使って新しい TrueCrypt ボリュームを作りたいなら、ここにチェックを入れてください。このオプションをチェックしなければ、トラベラーディスクの容量の節約になります。

暗号化アルゴリズム

TrueCrypt ボリュームは以下のアルゴリズムの一つで暗号化することができます。

アルゴリズム	設計者	キーサイズ (Bits)	ブロックサイズ (Bits)	モード
AES	J. Daemen, V. Rijmen	256	128	CBC
Blowfish	B. Schneier	448	64	CBC
CAST5	C. Adams, S. Tavares	128	64	CBC
Serpent	R. Anderson, E. Biham, L. Knudsen	256	128	CBC
Triple DES	IBM, NSA	3*56	64	Outer-CBC
Twofish	B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, N. Ferguson	256	128	CBC
AES-Blowfish		256;448	128;64	Inner-CBC
AES-Blowfish-Serpent		256;448;256	128;64;128	Inner-CBC
AES-Twofish		256;256	128	Outer-CBC
AES-Twofish-Serpent		256;256;256	128	Outer-CBC
Serpent-AES		256;256	128	Outer-CBC
Serpent-Twofish-AES		256;256;256	128	Outer-CBC
Twofish-Serpent		256;256	128	Outer-CBC

それぞれのセクターとボリュームで特有なランダム値は、IV として使われます。(詳細は「動作モード」を参照してください。

AES

Advanced Encryption Standard は FIPS（Federal Information Processing Standards 連邦情報処理規格）で承認された暗号アルゴリズム(Rijndael, designed by Joan Daemen and Vincent Rijmen, published in 1998)であり、アメリカ政府各部局、各機関で機密(または機密扱いでない)情報を保護するために[3]使われています。256-bit キー、14 ラウンドです。(AES-256, published in 2001)

2003 年 6 月に、NSA (US National Security Agency)が AES を分析、評価し、U.S. CNSS (Committee on National Security Systems)は[2]の中で AES-256(および AES-192)の強度は最高機密にいたるまでの機密扱いの情報を保護するのに充分であると発表しました。これは、Advanced Encryption Standard (AES)を使うか組み込むことで国家安全システムと国家安全情報に関連する Information Assurance の要求を満たすと考えるアメリカ政府各部局、各機関に適用されます。[2]

Blowfish

Bruce Schneier によって、1993 年に設計されました。特許はなく、ライセンス・フリーで、すべてのユーザーが入手可能です。TrueCrypt は Blowfish を 16 ラウンド、448-bit キーで使います。Blowfish は実装された暗号方式のうちで最速です。

CAST5

CAST5 または CAST-128 は Carlisle Adams と Stafford Tavares によって設計され、1997 年に発表されました。128-bit キー、64-bit ブロックです。この暗号化アルゴリズムは、アメリカ合衆国特許番号 5,511,123 [1]に記載されています。しかし、CAST5 は商業的でも非商業的にでも使用料はありません。[6] また、カナダ政府が機密(または機密扱いでない)情報を暗号化して保護する公式暗号化アルゴリズムの一つです。[17]

Serpent

Ross Anderson, Eli Biham, および Lars Knudsen によって設計され、1998 年に発表されました。256-bit キー、128-bit ブロックの Serpent は AES の最終形の一つです。これは Rijndael [4]より高度な安全性があるように見えるにもかかわらず、AES の推薦には選ばれませんでした。具体的には、Rijndael でも安全確保に充分であるのに対し、Serpent は高度な安全確保ができるように見えます。また、Rijndael はその数学的構造が将来攻撃対象となるかもしれないという、いくつかの批判を受けています。[4]
[5]において、Twofish チームは AES 最終形の安全係数の表を示しています。安全係数は、完全に暗号化するラウンド数をすでに破られた最大のラウンド数で割ったもので定義されます。だから、破られた暗号は最低の係数 1 ということになります。Serpent は AES 最終形の中で、(すべてのサポートされたキーサイズで)もっとも高い安全係数 3.56 を持ちます。Rijndael-256 の安全係数は 1.56 であり、Rijndael-128 は最終形の中で最低の安全係数 1.11 です。
これらの事実にもかかわらず、Rijndaelは安全性、速度、効率、実装のしやすさ[4]、柔軟性などのバランスのよさで、AESの中で適切な選択であると考えられています。最後のAES会議で、Rijndaelは 80 票、Serpentは 59 票、Twofishは 31 票、RC6 は 23 票、MARSは 13 票でした。[18, 19]*

Triple DES

1978 年に発表された Triple DES は IBM と NSA(1976 年)によって設計された 3 段階の DES 暗号です。Outer-CBC[16]と 3 つの独立した 56-bit キーが使われます(各段階で 1 つのキー)。[13] この暗号は非常に遅いことに注意してください。

* これは肯定的な票です。肯定的な票から否定的な票を引くと、次の結果となります。Rijndael: 76 票, Serpent: 52 票, Twofish: 10 票, RC6: -14 票, MARS: -70 票 [19]

Twofish

Bruce Schneier, David Wagner, John Kelsey, Niels Ferguson, Doug Whiting, Chris Hallによって設計され、1998年に発表されました。256-bitキー、128-bitブロックです。TwofishはAESの最終形の一つです。この暗号は、キーと独立したS-ボックスを使います。Twofishは、2^{128}(2 の 128 乗)の異なった暗号システムの集まりに見え、256-bitキーから派生する 128bitsがその集まりの中からの暗号システムの選択をコントロールします。[4] [23]の中で、Twofishチームは、キーから独立したSボックスが未知の攻撃に対する安全性を確保すると主張しています。[4]

AES-Blowfish

Inner-CBC モードで 2 つの暗号がカスケード(多段処理)されます。それぞれのセクターは、まずBlowfish(448-bit キー, 64-bit ブロック)で暗号化され、つぎに AES (256-bit キー, 128-bit ブロック)で暗号化されます。カスケードの暗号のそれぞれは、それ自身のキーを使います。それぞれのキーは相互に独立しています。(補足: ヘッダーキーは一つのパスワードから作られるものの、きちんと独立しています。「ヘッダーキーの生成、ソルト、反復回数」を参照) カスケードのそれぞれの暗号については、上記の個別解説を参照してください。

AES-Blowfish-Serpent

Inner-CBCモードで3つの暗号がカスケード(多段処理)されます。それぞれのセクターは、まずSerpent (256-bit キー, 128-bit ブロック)で暗号化され、次にBlowfish(448-bit キー, 64-bit ブロック)、最後にAES (256-bit キー, 128-bit ブロック)で暗号化されます。カスケードの暗号のそれぞれは、それ自身のキーを使います。それぞれのキーは相互に独立しています。(補足: ヘッダーキーは一つのパスワードから作られるものの、きちんと独立しています。「ヘッダーキーの生成、ソルト、反復回数」を参照) カスケードのそれぞれの暗号については、上記の個別解説を参照してください。

AES-Twofish

Outer-CBCモードで2つの暗号がカスケード(多段処理)されます。それぞれの128-bitブロックは、まずTwofish (256-bit キー)で暗号化され、つぎに AES (256-bit キー)で暗号化されます。カスケードの暗号のそれぞれは、それ自身のキーを使います。それぞれのキーは相互に独立しています。(補足: ヘッダーキーは一つのパスワードから作られるものの、きちんと独立しています。「ヘッダーキーの生成、ソルト、反復回数」を参照) カスケードのそれぞれの暗号については、上記の個別解説を参照してください。

AES-Twofish-Serpent

Outer-CBC モードで 3 つの暗号がカスケード(多段処理)されます。128-bit ブロックは、まず Serpent (256-bit キー, 128-bit ブロック)で暗号化され、次に Twofish (256-bit キー)、最後に AES (256-bit キー, 128-bit ブロック)で暗号化されます。カスケードの暗号のそれぞれは、それ自身のキーを使います。それぞれのキーは相互に独立しています。(補足: ヘッダーキーは一つのパスワードから作られるものの、きちんと独立しています。「ヘッダーキーの生成、ソルト、反復回数」を参照) カスケードのそれぞれの暗号については、上記の個別解説を参照してください。

Serpent-AES

Outer-CBC モードで 2 つの暗号がカスケード(多段処理)されます。それぞれの 128-bit ブロックは、まず AES (256-bit key)で暗号化され、つぎに Serpent (256-bit key)で暗号化されます。カスケードの暗号のそれぞれは、それ自身のキーを使います。それぞれのキーは相互に独立しています。(補足: ヘッダーキーは一つのパスワードから作られるものの、きちんと独立しています。「ヘッダーキーの生成、ソルト、反復回数」を参照) カスケードのそれぞれの暗号については、上記の個別解説を参照してください。

Serpent-Twofish-AES

Outer-CBC モードで 3 つの暗号がカスケード(多段処理)されます。128-bit ブロックは、まず AES (256-bit key)で暗号化され、次に Twofish (256-bit キー)、最後に Serpent (256-bit key)で暗号化されます。カスケードの暗号のそれぞれは、それ自身のキーを使います。それぞれのキーは相互に独立しています。(補足: ヘッダーキーは一つのパスワードから作られるものの、きちんと独立しています。「ヘッダーキーの生成、ソルト、反復回数」を参照) カスケードのそれぞれの暗号については、上記の個別解説を参照してください。

Twofish-Serpent

Outer-CBC モードで 2 つの暗号がカスケード(多段処理)されます。それぞれの 128-bit ブロックは、まず Serpent (256-bit key)で暗号化され、つぎに Twofish (256-bit key)で暗号化されます。カスケードの暗号のそれぞれは、それ自身のキーを使います。それぞれのキーは相互に独立しています。(補足: ヘッダーキーは一つのパスワードから作られるものの、きちんと独立しています。「ヘッダーキーの生成、ソルト、反復回数」を参照) カスケードのそれぞれの暗号については、上記の個別解説を参照してください。

動作対象オペレーションシステム

TrueCrypt は以下のオペレーションシステムで動作します。

- Windows XP
- Windows 2000
- Windows(Server) 2003

(すくなくとも ServicePack 1 以降の) Windows XP またはその後継で TrueCrypt に最適な環境のものを推薦します。

Windows 64-Bit 版は現在のところ、サポートしていません。

注: TrueCrypt は Windows "Longhorn" (Windows XP 後継のベータ版) でも動きます。

コマンドラインの使い方

/help	*/?*	コマンドラインのヘルプを表示
/volume	*/v*	TrueCrypt ボリュームのファイルとパスの名前を表示。ハードディスクのパーティションをマウントする場合の例は /v ¥Device¥Harddisk1¥Partition3(パーティションのパスを決めるには、TrueCrypt を起動して「デバイスの選択」をクリックしてください。

/letter	*/l*	ボリュームをマウントするドライブレター。/l が省略され /a が指定されている場合には最初の空きドライブレターを使います。
/explore	*/e*	ボリュームが正常にマウントされると、ボリュームを開きます。
/beep	*/b*	ボリュームが正常にマウントまたはアンマウントされるとビープを鳴らします。
/auto	*/a*	パラメータが指定されていなければ、ボリュームを自動マウントします。デバイスがパラメータとして指定されていれば、使用可能なデバイス/パーティション型 TrueCrypt ボリュームを自動マウントします。
/dismount	*/d*	ドライブレターが指定されていれば、そのボリュームをアンマウントします。指定がなければ、現在マウントされている TrueCrypt ボリュームをすべてアンマウントします。
/force	*/f*	アンマウントしたいボリュームがシステムやアプリケーションでつかわれている場合に強制的にアンマウントします。また、そのようなボリュームを共有モード(排他アクセスなし)でマウントをします。
/cache	*/c*	y はパスワードの記憶を有効に、n は無効にします。パスワード記憶を無効にしても、すでに記憶されているものは消去されないことに注意してください。
/history	*/h*	y は履歴を有効に、n は無効にします。
/wipecache	*/w*	ドライバーのメモリに記憶しているすべてのパスワードを消去します。
/password	*/p*	ボリュームのパスワード。パスワードが空白を含む場合には引用符で囲んでください。(/p “My Password” のように) ***警告:*** *この方法でパスワードを入力することは安全上の問題があります。特に、暗号化されていないバッチファイルに記述したり、暗号化されていないコマンドプロンプトの履歴がディスクに記録される場合などです。かわりに /q /a を使うことを検討してください。*
/quiet	*/q*	控えめな動作をします。/auto とともに指定され、記憶されたパスワードが正しくない場椎には、パスワード入力を求めるプロンプトが表示されます。(TrueCrypt メイン・ウィンドーは開きません) これは、マルチユーザー環境でのプライバシー・レベルを向上させます。このモードでは、プログラム設定はレジストリから読まれたり書かれたりすることはありません。
/mountoption	*/m*	ro : 読み取り専用としてマウント。rm : リムーバブルメディアとしてマウント。

文法

truecrypt [/v volume] [/d [letter]] [/l letter] [/m {rm|ro}] [/e] [/b] [/p password] [/h {y|n}] [/q] [/c {y|n}] [/w] [/a [devices]] [/f]

パラメータの順番は重要ではありません。パラメータとその値との間の空白はあってもなくてもかまいません。

例

’myvolume.tc’というボリュームを‘MyPassword’というパスワードで、ドライブレター X にマウントし、正常にマウントできたらボリュームをひらき、ビープを鳴らします。マウントは自動的に実行されます。

truecrypt /v myvolume.tc /lx /a /p MyPassword /e /b

’myvolume.tc’というボリュームを最初の空いたドライブレターにマウントし、パスワードプロンプトでパスワードを入力します。(メインプログラム・ウィンドーは表示されません)

truecrypt /v myvolume.tc /a /q

安全のための予防策

ここでは機密データの安全性に影響するいくつかの項目について述べます。(ほとんどは直接には TrueCrypt とは関係しません) すべての危険性について網羅することはできないことを、ご了承ください。残念ながら非常に多くの種類の危険があり、すべてを解説しようとするとあまりに膨大になってしまうためです。

スワップファイル

ページングファイルとも呼ばれます。Windows は(通常ハードディスクに置かれている)このファイルを、メモリに入らないプログラムやデータファイルを保持するために使います。ということは、メモリ上だけにあると信じている機密データが実際には知らないうちに Windows によって暗号化もされずにディスクに書かれているということです。
TrueCrypt はパスワード、暗号化キー、IV、および他のボリューム用機密データをページングされないメモリに置くので、ページングファイルへデータが洩れることはありません。しかし TrueCrypt は、RAM 上に開かれた機密ファイルが暗号化されない状態でスワップに保存されることを防ぐことはできません。(TrueCrypt ボリュームのファイルを開くと、そのファイルの内容は暗号化されていない状態で RAM に置かれます)
ですから、Windows XP ユーザーには、スワップファイル機能を無効にすること、少なくとも機密データを使うセッションの間だけでも無効にすることを強くおすすめします。これをするにはデスクトップかスタートメニューのマイコンピュータ・アイコンの上で右クリックし、*プロパティ-> 詳細設定-> パフォーマンス-> 設定-> 詳細設定-> 仮想メモリ> 変更-> ページングファイルなし-> 設定-> OK* としてください。
知る限りでは、残念ながらWindows 2000 ではこの方法では完全に無効にはできません。Windows 2000 ユーザーには、コンピュータをシャットダウンするつどにページングファイルをクリアするようセキュリティの設定を変更することをおすすめします。(詳細はWindowsのマニュアルまたは www.microsoft.com を参照してください)

ハイバネーション・モード

コンピュータがハイバネーション・モードに入るとき、システムメモリやオープンされたファイルの内容はハードディスクに書き出されます。TrueCrypt は RAM 上に開かれた機密ファイルが暗号化されない状態でハイバネーション・ファイルに保存されることを防ぐことはできません。(TrueCrypt ボリュームのファイルを開くと、そのファイルの内容は暗号化されていない状態で RAM に置かれます) ですから、少なくとも機密データを使うセッションの間だけでもハイバネーション機能を無効にするか、ハイバネーションの起動を抑止することを強くおすすめします。

マルチユーザー環境

マウントされた TrueCrypt ボリュームの内容はすべてのログオンしたユーザーには見えてしまうということを忘れないでください。(NTFS ではファイルの許可情報の設定で、このようなことを防ぐことは可能です) また、ユーザー切替やログオフは正常にマウントされた TrueCrypt ボリュームをアンマウントしないことに注意してください。(システムを再起動する場合には、すべてのマウントされた TrueCrypt ボリュームはアンマウントされます)

データの破損

ハードウェアやソフトウェアのエラーや誤動作で、TrueCrypt ボリュームのファイルが破損することもありえます。ですから、重要ファイルは定期的にバックアップをとることをすすめます。(もちろん、TrueCrypt ボリュームに記録されたデータにかぎらず、すべての重要なデータについて言えることです)

非互換性

現在のところ、特に非互換性について記載することはありません。

既知の問題と制限

- ネットワーク経由でボリュームを扱うことはサポートされていません(マウントされたTrueCryptボリュームの内容をネットワークで共有することは可能ですが、ネットワークの向こうにある TrueCrypt ボリュームをマウントすることはできません)

- TrueCrypt で暗号化されたフロッピーディスク: フロッピーディスクが排出され他のディスクが挿入されると、ゴミが書かれたり読まれたりしてデータが破損するかもしれません。これはフロッピーディスクをまるごとボリュームとして扱う場合で、フロッピーディスク上のファイル形式コンテナの場合ではありません)

問題が起こったら

ここでは TrueCrypt を使っていて遭遇するかもしれない一般的な問題への解決策を提示します。ここにない問題であれば、次のところに記載があるかもしれません。

非互換性
既知の問題と制限
FAQ(よくある質問)

問題:

ボリュームは正常にマウントされ、TrueCrypt ではそのボリュームはマウントされていると表示されているにもかかわらず、Windows の Explorer からボリュームにアクセスできない。(マイコンピュータなどにも表示されない)

想定される原因:

Windows Explorer の問題です。

対応案:

*ツール -> ドライブレターの更新*をクリックしてください。これでだめなら、Windows Explorer を再起動してください。(たとえば、ログオフして再度ログオンするなど)

問題:

隠しボリュームを作ろうとしたら、作成可能な最大サイズが予想外に小さい。(外殻ボリュームにはこれよりずっと大きい空き容量があるのですが)

想定される原因:

ファイルの断片化(フラグメンテーション)

または

クラスタサイズが小さすぎるところに、外殻ボリュームのルートディレクトリに置いたフォルダーやファイルが多すぎるということが考えられます。

対応案:

外殻ボリュームにデフラグをかける。(マイコンピュータのそのドライブレターを右クリック、プロパティをクリック、ツール・タブを選択、「最適化する」をクリック)　ボリュームのデフラグが終わったら、もう一度隠しボリューム作成を試してください。

これでもだめなら、外殻ボリュームのすべてのファイルとフォルダーを Shift+Delete を押すことで削除してください。フォーマットで消してはいけません。(事前に Recycle Bin と SystemRestore を無効にすることを忘れないでください)　そして、完全に空になった外殻ボリュームに隠しボリュームを作成してみてください。(テスト目的だけです)　それでも隠しボリュームの可能な最大サイズが変わらなければ、問題は拡張ルートディレクトリにありそうです。もし(ウィザードの最終ステップで)クラスタサイズを初期値のままにしなかったなら、こんどはクラスタサイズを初期値のままにして外殻ボリュームをフォーマットしなおしてください。

さらにこれでもだめなら、外殻ボリュームを再フォーマットして前回より少ないファイルやフォルダーをルートに置いてください。それでだめなら、再フォーマットしてルートのファイルやフォルダーを減らすことを繰り返してください。やってられないとか、効果なしなら、解決するまで外殻ボリュームをクラスタサイズを大きくしながら再フォーマットを繰り返してください。もし、なぜクラスタサイズがそんなに大きいのかと聞かれたら、より高性能(高速)を目指したからと答えてください。(「クラスタサイズ」を参照)

問題:

パーティション/デバイスを暗号化しようとすると、TrueCrypt ボリュームウィザードから使用中だというメッセージが出ます。

対応案:

OS のブートパーティションを暗号化しようとはしていませんね？(TrueCrypt は、これはサポートしていません)
そうでなければ、そのパーティション/デバイスを何らかの形で使うプログラム(たとえば、アンチウィルスなど)を停止、アンインストールなどしてください。これでもだめなら、デスクトップのマイコンピュータ・アイコンを右クリックして *管理* -> *記憶域* -> *ディスクの管理*を選んでください。そこで暗号化したいパーティションをクリックし、*ドライブレターの変更*をクリックし、*ドライブ文字とパスの変更*をクリック、*削除*をクリックして *OK* としてください。
最後にシステムを再起動してください。

問題:

隠しボリュームを作成しようとすると、ウィザードが外殻ボリュームをロックできないと言ってきます。

想定される原因:

外殻ボリュームのファイルを何かのアプリケーションが開いています。

対応案:

外殻ボリュームのファイルを使うアプリケーションをすべて閉じてください。それでもだめなら、アンチウィルスを停止するかアンインストールして試してください。

問題:

以下のどれかが発生:

1. TrueCrypt ボリュームをマウントできない。
2. NTFS TrueCrypt ボリュームを作成できない。

さらに、エラーメッセージが出る: *他のプロセスで使用中のため、プロセスはファイルにアクセスできません。*

想定される原因:

他のアプリケーションが干渉している可能性があります。これは TrueCrypt のバグではありません。OS が他のアプリケーションが排他アクセスのためデバイスをロックしていると TrueCrypt へ通知しています。(だから TrueCrypt はデバイスにアクセスできないわけです)

対応案:

干渉するアプリケーションを停止またはアンインストールすることで、通常は解決します。アンチウィルスやディスク管理ツールなどがこの例です。

FAQ (よくある質問)

TrueCrypt FAQの最新版はhttp://truecrypt.sourceforge.net/faq.php で入手できます。(英語版)

Q: パスワードを忘れてしまいました。TrueCrypt ボリュームのファイルを復元する方法はありませんか？

A: TrueCrypt は正しいパスワードまたは暗号化に使ったキーなしで、暗号化されたデータを部分的でも完全にでも復元する機能はまったく持っていません。復元するたった一つの方法は暗号を破ることですが、パスワードの質や長さ、キーのサイズ、ソフトやハードの効率性、その他の要素によって、数千年、数百万年かかるかもしれません。

Q: TrueCrypt はパスワードをディスクに保存しますか？

A: いいえ。

Q: パスワードのハッシュはどこかに保存されますか？

A: いいえ。

Q: HMAC-SHA-1 か HMAC-RIPEMD-160 を使うとき、キーサイズは 160 ビットに制限されているのですか？

A: いいえ。TrueCrypt は(HMAC アルゴリズムだけではなく)ハッシュ関数の出力を直接暗号化キーとして使うことはありません。詳細は*「ヘッダーキーの取得、ソルト、反復回数」*を参照してください。

Q: TrueCrypt が扱える最大ボリュームサイズはどのくらいですか？

A: TrueCrypt ボリュームは 9223372036 GB までを扱えます。しかし現実には以下の問題があります:
コンテナが保管されるファイルシステムの制限、そのファイルシステムでのコンテナ自身の制限、ハードウェアや OS の制限など。
FAT32 ファイルシステムに置かれるファイル形式のコンテナは、4GB を越えることができません。(もっと大きいボリュームが必要な場合は、NTFS ファイルシステムに置くか、ファイル形式ボリュームのかわりに暗号化パーティションを使ってください) また、どんな FAT32 ボリュームでも(暗号化されているかどうかにかかわらず)2048GB を越えることはできません。(もっと大きいボリュームが必要な場合は、NTFS でフォーマットしてください)

Q: どの暗号方式がもっとも安全ですか？

A: 残念ですが、この質問には回答できません。しかし、TrueCrypt に実装されているすべての暗号化方式はよく知られ、信頼されているものばかりです。脆弱な方式は TrueCrypt には実装していません。

Q: TrueCrypt ボリュームにアプリケーションをインストールし、動かすことができますか？

A: はい。

Q: TrueCrypt はどのようにして正しいパスワードが入力されたかを判断しているのですか？

A: 「技術解説」の「暗号化の仕組み」を参照してください。

Q: TrueCrypt はハードウェア/ソフトウェア レイドとダイナミックボリュームをサポートしていますか？

A: はい、どちらもサポートしています。ダイナミックボリュームを TrueCrypt ボリュームとしてフォーマットする場合には、(Windows のディスク管理ツールを使って)ダイナミックボリュームを作成したあと、システムを再起動して、TrueCrypt ボリューム作成ウィザードの「デバイス選択」に目的のボリュームが表示され、選択できるようにすることを忘れないようにしてください。「デバイス選択」ウィンドーで、ダイナミックボリュームは単一のデバイスとしては表示されません。そのかわり、ダイナミックボリュームを構成するすべてのボリュームが表示され、ダイナミックディスク全体をフォーマットするために、どれか一つを選択することができます。

Q: CD や DVD に保管された TrueCrypt コンテナをマウントできますか？

A: はい、できます。しかし、Windows2000 で読み出し専用メディア(CD/DVD 他)にある TrueCrypt ボリュームをマウントする場合には、TrueCrypt ボリュームを FAT でフォーマットしなくてはならないことを憶えておいてください。(Windows2000 では読み取り専用メディアの NTFS ファイルシステムはマウントできません)

Q: TrueCrypt パーティションをフォーマットすると、どうなりますか？

A: この FAQ の「暗号化ボリュームのファイルシステムを変更できますか？」を参照してください。

Q: 暗号化ボリュームのファイルシステムを変更できますか？

A: マウントされていれば、可能です。TrueCrypt ボリュームは FAT12, FAT16, FAT32, または NTFS でフォーマットすることができます。ボリュームは普通のボリュームと同じように扱うことができるので、マイコンピュータなどでデバイスを右クリックし、フォーマットを選んでください。ボリュームの内容は失われますが、ボリームは暗号化された状態のままになります。もし、パーティション形式の TrueCrypt ボリュームがマウントされていないときに、そのパーティションをフォーマットすると、ボリュームは破壊され、パーティションは暗号化された状態ではなくなり、空となります。

Q: 隠しボリュームのパスワードを変更できますか？

A: はい。パスワード変更ダイアログは標準ボリュームにも隠しボリュームにも機能します。*ボリュームパスワード変更ダイアログの現在のパスワード*に隠しボリュームのパスワードを入力してください。

注: TrueCrypt は最初に標準ボリュームヘッダーを復号しようとします。これに失敗するとその中に隠しボリュームがあると想定し、隠しボリュームのヘッダーがあると想定される位置を復号しようとします。これが成功するとパスワード変更は隠しボリュームに対して適用されることになります。(どちらの試みも*現在のパスワード*に入力されたパスワードを使います)

Q: TrueCrypt パーティションを通常のものに戻すには、どうすればいいですか？

A: パーティション形式の TrueCrypt ボリュームがマウントされていないときに、そのパーティションをフォーマットすると、ボリュームは破壊され、パーティションは暗号化された状態ではなくなり、空となります。
このような場合、TrueCrypt ボリュームの内容はすべて失われることに注意してください。

Q: 2GB 以上の TrueCrypt コンテナをどうやって DVD に焼くのですか？

A: あなたが使っている DVD 作成ソフトで DVD のフォーマットを選択できるはずです。そこで、UDF フォーマットを選んでください。(ISO フォーマットは 2GB 以上をサポートしていません)

Q: Windows の *ファイルセレクタ* がマウントした最後のコンテナを記憶しています。防止できますか？

A: それを防止するにはいくつかの方法があります。一つは Windows レジストリを適切に編集することです。(詳細は Windows のドキュメントを参照してください) もう一つの方法は TrueCrypt コンテナを選ぶのにファイルセレクタを使わないことです。かわりに、そのアイコンを TrueCrypt.exe アイコンか TrueCrypt ウィンドーへドラッグしてください。TrueCrypt.exe アイコンへドラッグした場合には、TrueCrypt は自動で起動します。

Q: 現在保存しているデータを失わずに、パーティションを暗号化できますか？

A: 残念ですが、TrueCrypt ではこのようなことはできませんし、今後も用意するつもりもありません。
われわれは、この方法は安全性に問題があると考えているためです。(また、安全にこのようなことをしようとすると、暗号化にかかる時間が耐えられないほど長くなってしまいます) 詳細は[21]を参照してください。

Q: TrueCrypt ボリュームの内容に対して、chkdsk や Defrag といったツールを使うことはできますか？

A: はい。TrueCrypt ボリュームは本物の物理的なディスクと同じに扱うことができますから、どんなファイルシステムのチェックや修復、デフラグのツールでもマウントされた TrueCrypt ボリュームに対して使うことができます。

Q: OS の起動パーティションを暗号化できますか？

A: いいえ、TrueCryptはこのようなことはできません。しかし、OSが存在するボリュームをリードオンリーにして(レジストリ、臨時ファイル、他はRAMに保持されます)、情報漏洩を防止し、トロイの木馬を仕掛けられないことを保証する方法があります。方法の一つはBartPE(Bart's Preinstalled Environment バートのプリインストール環境)を使うことです。これは、基本的にWindowsそのものをCD/DVDに格納し(レジストリ、臨時ファイル、他はRAMに保持されます)、そこからWindowsを起動するというものです。フリーウェアである*Bart's PE Builder* はWindowsインストールCDをBartPEに変換することができます。
TrueCrypt 3.1 以降を使っているなら、BartPE の TrueCrypt プラグインは必要ありません。単に BartPE のもとで、BartPE ディスクからか、TrueCrypt システムファイル(TrueCrypt.exe, TrueCrypt.sys 他)がある他の場所から TrueCrypt をトラベラーモードで起動するだけです。BartPE を置くのは CD か DVD か、一度だけ書き込み可能で、いくらでも読み出しできるもの(たとえば CD-R)にするべきです。CD-RW のような書き換え可能なディスクでは、誰かに内容を書き換えられてしまう可能性があるためです。

Q: TrueCrypt ボリュームの中の TrueCrypt ボリュームをマウントすることはできますか？

A: はい、TrueCrypt ボリュームは無制限に入れ子にできます。

Q: TrueCrypt と他の自動即時暗号化ツールを同じシステムで併用できますか？

A: TrueCrypt と他の自動即時暗号化ツールを併用することで問題が起きるとも起きないとも聞いていません。

Q: TrueCrypt パーティションのサイズを変更できますか？

A: 残念ですが、こういったことはできません。PartitionMagic のようなプログラムで TrueCrypt パーティションのサイズを変更すると、多くの場合はデータを壊すことになるでしょう。

Q: 管理者権限がなくても TrueCrypt を使うことはできますか？

A: 管理者権限がなくても TrueCrypt ボリュームをマウントしたり、アンマウントしたり、作成したりはできます。しかし、パーティションを暗号化したりフォーマットしたりはできません。(ファイルコンテナを作ることができるだけです)。また、NTFS ボリュームを作ったり、TrueCrypt をインストールしたりアンインストールしたりもできません。

Q: なぜ好きな(暗号化アルゴリズムの組み合わせの)カスケードを作れないのですか？

A: 理由は、どの暗号化アルゴリズム(および動作モード)が TrueCrypt ボリュームを暗号化したのか不明なためです。そのボリュームに適切な暗号化アルゴリズムはトライアル・アンド・エラーの過程を通じて決定されます。もし、任意の組み合わせのカスケードを認めると、マウントするときに試さなくてはならない暗号化アルゴリズムの数が膨大になってしまいます。そうなると、特に遅い PC ではマウントするためにかかる時間が使用に耐えないほど長くかかることになります。

Q: TrueCrypt ボリュームの一部が破損したら、どうなりますか？

A: それぞれのセクター(セクターは 512byte)のデータがつながってしまい(*「動作モード」*参照)、ブロックが破損するとセクター内のブロックも破損します。(ブロックサイズは暗号化アルゴリズムによって、8 か 16bytes です) ボリュームヘッダーが破損すると、多くの場合はそのボリュームをマウントできなくなります。

Q: TrueCrypt ボリュームが断片化しているかどうかをかまわず、マウントできますか？

A: はい。

Q: TrueCrypt コンテナをコピーする前に、コンピュータを再起動させる必要がありますか？

A: いいえ、必要ありません。

Q: 既存のコンテナを複製することで、新しいコンテナを作っても安全ですか？

A: 新しい TrueCrypt コンテナを作る場合は、つねにボリューム作成ウィザードを使ってください。もし、コンテナをコピーして両方を使うと、両方に異なったデータが入ることになり暗号解析の手がかりになるかもしれません。なぜなら、両方のボリュームが同じキー(IV、ホワイトニング値 等)を持つためです。

Q: TrueCrypt の空き領域やファイルなどを完全消去しなくてはなりませんか？

[完全消去 -機密データを上書きして復活不可能にすること]
A: 敵対者が(あなたにパスワードを明かさせるなど)ボリュームを復号できると信じるなら、そうしてください。そうでなければ必要ありません。ボリュームはまるごと暗号化されていますから。

Q: TrueCrypt は E4M とどう関係しているのですか？

A: TrueCrypt 1.0 は E4M 2.02a に由来していました。E4M と TrueCrypt との差異については、「バージョン履歴」を参照してください。

Q: TrueCrypt はずっとこのままオープンソースでフリーなのですか？

A: はい、そうです。商業版は計画していませんし、そうもならないでしょう。私たちはオープンソースでフリーなセキュリティソフトウェアに信頼をおいています。
補足: TrueCrypt を有料でクローズドソースとして配布しようとしている人たちがいることは知っています。私たちはそのような人たちとは手を組みません。

TrueCrypt のアンインストール

TrueCrypt をアンインストールするには、Windows のコントロールパネルを開き、「アプリケーションの追加と削除」で TrueCrypt を選び、「変更と削除」をクリックしてください。

通常は、アンインストーラ(WindowsPath¥TrueCryptSetup.exe)以外のすべての TrueCrypt ファイルとレジストリの関連データの大部分が削除されます。

TrueCrypt ボリュームは削除されません。TrueCrypt を再インストールすれば、そのボリュームをまたマウントできます。

TrueCrypt システムファイル

WindowsPath¥TrueCryptSetup.exe (アンインストーラ)

WindowsPath¥SYSTEM32¥DRIVERS¥truecrypt.sys (ドライバー)

Windows パスをあなたのシステムの Windows がインストールされたパス(たとえば C:¥WINDOWS)に置き換えてください。

技術解説

表記法

C	暗号テキストブロック
$D_K()$	復号キー K を使う復号アルゴリズム
$E_K()$	暗号化キー K を使う暗号化アルゴリズム
H()	ハッシュ関数 (e.g., RIPEMD-160)
i	n-bit ブロックのブロックインデックス; n は状況による
K	暗号化/復号キー
m	64-bit ブロックのブロックインデックス
n	128-bit ブロックのブロックインデックス
P	プレーンテキストブロック
R	乱数プール
W	ホワイトニング値
^	排他的論理和 (XOR)
⊕	加算して2^nで割った余り(Modulo 2^n addition)。n が左のオペランドと結果のビットサイズ。(もし、左のオペランドが1-bit値で、右のオペランドが2-bit値の場合: 1 ⊕ 0 = 1; 1 ⊕ 1 = 0; 1 ⊕ 2 = 1; 1 ⊕ 3 = 0; 0 ⊕ 0 = 0; 0 ⊕ 1 = 1; 0 ⊕ 2 = 0; 0 ⊕ 3 = 1)
||	連結

暗号化の仕組み

TrueCrypt ボリュームをマウントするとき(パスワードが記憶されていないと仮定して)、次のステップが実行されます。

1. ボリュームの最初の 512 バイト(標準ボリュームのヘッダー)が RAM に読み込まれます。その最初の 64 ビットがソルトです。(「TrueCrypt ボリュームフォーマット仕様」を参照)

2. ボリュームの最後から 1536 バイトの位置から 512 バイトが RAM に読み込まれます。(「TrueCrypt ボリュームフォーマット仕様」を参照) もしそのボリュームに隠しファイルがあれば、そのヘッダーを読み込んだことになります。(隠しボリュームがあるかないかは、このデータを復号できるかどうかで決まります)

3. TrueCryptは(1)と(2)で読み込んだ標準ボリュームのヘッダーと隠しボリュームのヘッダー (かもしれないデータ)を復号しようとします。復号の過程で使われたり生成されたりしたデータはRAMに保持されます。(TrueCryptはこれらをけっしてディスクに保存しません) 次のパラメータは未知*で、試行錯誤で決定していきます。(以下の可能な組み合わせをすべて試します)

   a. ヘッダーキー生成に使われる PRF(PKCS #5 v2.0 に規定。「ヘッダーキーの生成、ソルト、反復回数」を参照) これは以下のどちらかになります: HMAC-SHA-1, HMAC-RIPEMD-160。ユーザーが入力したパスワードや(1)か(2)で読み込まれたソルトはヘッダーキー生成関数へ渡され、一連の値(「ヘッダーキーの生成、ソルト、反復回数」を参照)が作られます。そしてそれから、ヘッダー暗号化キーとIV(ボリュームヘッダーを復号するのに使われる)が生成されます。
   b. 暗号化アルゴリズム: AES-256, Blowfish, CAST5, Serpent, Triple DES, Twofish
   c. 使う暗号の数(単体の暗号かカスケードか)
   d. 動作モード: CBC, inner-CBC, outer-CBC
   e. ブロックサイズ
   f. キーサイズ

4. 復号データの最初の 4 バイトが"TRUE"という文字列であり、復号されたデータ(ボリュームヘッダー)の最後の 256 バイトの CRC-32 チェックサムが復号データの 8 番目のバイトの値と一致したなら、復号が成功したと判断します。(この値は暗号化されているので、敵対者にはわかりません。「TrueCrypt ボリュームフォーマット仕様」を参照) もし、これらの条件が満たされなければマウントは中止されます。(間違ったパスワードか、ボリュームが破損しているか、TrueCrypt ボリュームではないということです)

5. これで正しいパスワード、適切な暗号化アルゴリズム、モード、キーサイズ、ブロックサイズ、正しいヘッダーキー生成アルゴリズムがわかった(あるいは非常に高い可能性でわかったと仮定できる)ことになります。また、隠しボリュームをマウントしようとしているのかどうかもわかりました。復号されたボリュームヘッダーにあるボリュームを開くのに必要な最小のプログラムバージョンもチェックされます。もしボリュームをマウントしたプログラムのバージョンより古かったら、マウントは中止されます。

6. 暗号化ルーチンは復号されたボリュームヘッダーから得られたマスターキー†とIVで再初期化されます。このキーはボリュームヘッダー領域をのぞく、ボリュームのどのセクターでも復号するのに使うことができます。(ボリュームヘッダー領域は、ヘッダーキーで暗号化されます)

「動作モード」、「ホワイトニング」、「ヘッダーキーの生成、ソルトおよび反復回数」も参照してください。

* これらのパラメータは、攻撃の困難さを強化するために秘密にされているのではなく、TrueCryptボリュームであるかどうかを事前に知ることができないためです。(単なるランダムデータと区別がつかない) ボリュームヘッダーにこれらのパラメータを格納しておくと、こうはなりません。

† マスターキーはボリューム作成のときに生成され、あとで変更することはできません。ボリュームのパスワード変更は、新しいパスワードから生成される新しいヘッダーキーでボリュームヘッダーを再暗号化することで実施されます。

動作モード

次の表は TrueCrypt に実装されているすべてのアルゴリズムと、それらの動作モードです。

暗号化アルゴリズム	動作モード	動作モード詳細
AES*	CBC	$C_i = E_K(P_i \oplus C_{i-1})$; C_0 is IV.
AES-Blowfish (*E*2) (*E*1)	Inner-CBC	$C_n = E2_{K2}((S_m \parallel S_{m+1}) \oplus C_{n-1})$; $S_m = E1_{K1}(P_m \oplus S_{m-1})$; C_0 and S_0 are IVs.
AES-Blowfish-Serpent (*E*3) (*E*2) (*E*1)	Inner-CBC	$C_n = E3_{K3}((S_m \parallel S_{m+1}) \oplus C_{n-1})$; $S_m = E2_{K2}(T_m \oplus S_{m-1})$; $T_m \parallel T_{m+1} = Q_n = E1_{K1}(P_n \oplus Q_{n-1})$; C_0, S_0, and Q_0 are IVs.†
AES-Twofish (*E*2) (*E*1)	Outer-CBC	$C_i = E2_{K2}(E1_{K1}(P_i \oplus C_{i-1}))$; C_0 is IV.
AES-Twofish-Serpent (*E*3) (*E*2) (*E*1)	Outer-CBC CBC	$C_i = E3_{K3}(E2_{K2}(E1_{K1}(P_i \oplus C_{i-1})))$; C_0 is IV.
Blowfish		$C_i = E_K(P_i \oplus C_{i-1})$; C_0 is IV.
CAST5	CBC	$C_i = E_K(P_i \oplus C_{i-1})$; C_0 is IV.
Serpent	CBC	$C_i = E_K(P_i \oplus C_{i-1})$; C_0 is IV.
Serpent-AES (*E*2) (*E*1)	Outer-CBC	$C_i = E2_{K2}(E1_{K1}(P_i \oplus C_{i-1}))$; C_0 is IV.
Serpent-Twofish-AES (*E*3) (*E*2) (*E*1)	Outer-CBC	$C_i = E3_{K3}(E2_{K2}(E1_{K1}(P_i \oplus C_{i-1})))$; C_0 is IV.
Triple DES	Outer-CBC	$C_i = E_{K3}(D_{K2}(E_{K1}(P_i \oplus C_{i-1})))$; C_0 is IV.
Twofish	CBC	$C_i = E_K(P_i \oplus C_{i-1})$; C_0 is IV.
Twofish-Serpent (*E*2) (*E*1)	Outer-CBC	$C_i = E2_{K2}(E1_{K1}(P_i \oplus C_{i-1}))$; C_0 is IV.

* この表では、”AES”は”AES-256”を意味しています。

† $T_m \parallel T_{m+1} = Q_n$ は 128-bit ブロックQ_n が 2 つの 64-bit ブロックT_m と T_{m+1} に分割されるということです。

IV(initialisation vector 初期化ベクター)はつねに(敵対者にはわからない)ランダムな値であり、それぞれのセクター*とボリュームに特有のものになります。この値は以下の手順で生成されます。

1. 復号されたボリュームヘッダーの 256-263(128-bit ブロック暗号では 256-271)が取得されます。(「TrueCrypt ボリュームフォーマット仕様」を参照) カスケードの暗号が一つ以上の IV を使うなら、引き続いて追加のバイトが取得されます。(たとえば、カスケードに inner-CBC モードの 3 つの 128-bit ブロック暗号があるとすると、256-271 が最初の IV として取得され、272-287 が二番目の IV、288-303 が三番目の IV として取得されます)

2. (1)で取得されたデータは 64-bit セクター番号と XOR されます。(それぞれのセクターの大きさは 512 バイトで、0 から始まる番号がつけられます) 128-bit ブロック暗号の場合には、(1)で得られた 128-bit 値の上位と下位の 64-bit ワードが対応する値と XOR されます) 結果として得られる 64-bit 値(128-bit ブロック暗号では 128-bit)が IV となります。

   つまり、
   128-bit ブロック暗号では:
   T_1 = (1)で得られた値の上位 64-bit ワード
   T_2 = (1)で得られた値の下位 64-bit ワード
   S = セクター番号*(64-bit 整数)*
   $IV = (T_1 \wedge S) \| (T_2 \wedge S)$

   64-bit ブロック暗号では:
   T = (1)で得られた値
   S = セクター番号*(64-bit 整数)*
   $IV = T \wedge S$

   補足: ステップ(1)は、ボリュームがマウントされた直後に 1 回だけ実行されます。取得された値は RAM に残ります。

カスケードの暗号は、おたがいに独立したキーを使います。(補足: ヘッダーキーは一つのパスワードから作られるものの、きちんと独立しています。「ヘッダーキーの生成、ソルト、反復回数」を参照)

ホワイトニング

プレーンテキストと暗号テキストのペアを得るのは、さらに難しくなります。[16] ホワイトニングと呼ばれるものに似た以下の手法が使われます。(セクターが復号されたあと)セクターの各 8 バイトがセクターやボリュームごとに特有の(敵対者にはわからない)値と XOR されます。この値は下記の方法で生成されます。

1. 復号されたボリュームヘッダーの 264-271(128-bitブロック暗号では 272-279)が取得†されます。(「TrueCryptボリュームフォーマット仕様」を参照)

2. 復号されたボリュームヘッダーの 272-279(128-bit ブロック暗号では 280-287)が取得されます。

3. (1)で取得されたデータは 64-bit セクター番号と XOR されます。(それぞれのセクターの大きさは 512 バイトで、0 から始まる番号がつけられます)

4. (2)で取得されたデータは 64-bit セクター番号と XOR されます。

*それぞれのセクターの大きさは 512 バイトで、0 から始まる番号がつけられます

†必要なデータは、ボリュームがマウントされた直後にボリュームヘッダーから取得され、ボリュームがアンマウントされるまでRAM に保持されます。

5. (3)の結果の値の最初の 8 バイトの 32-bit CRC-32 の値が計算されます。

6. (3)の結果の値の二番目の 8 バイトの 32-bit CRC-32 の値が計算されます。

7. (4)の結果の値の最初の 8 バイトの 32-bit CRC-32 の値が計算されます。

8. (4)の結果の値の二番目の 8 バイトの 32-bit CRC-32 の値が計算されます。

9. (5)で計算された値と(8)で計算された値の XOR をとります。

10. (6)で計算された値と(7)で計算された値の XOR をとります。

11. (9)で計算された 32-bit 値が 64-bit ホワイトニング値の上位 32-bit ワードに、(10)で計算された 32-bit 値が下位 32-bit ワードに、書き込まれます。

補足: ステップ 5-8 は個々のホワイトニング値間の影響を少なくするため、実行されます。

要約:

T_1 = (1)で得られた 64-bit 値
T_2 = (2)で得られた 64-bit 値
S = セクター番号 *(64-bit 整数)*
$T_1 = T_1 \wedge S$
$T_2 = T_2 \wedge S$
Q_1 = T_1 の上位 32-bit ワード
Q_2 = T_1 の下位 32-bit ワード
Q_3 = T_2 の上位 32-bit ワード
Q_4 = T_2 の下位 32-bit ワード
$W = (\mathrm{CRC32}(Q_1) \wedge \mathrm{CRC32}(Q_4)) \,||\, (\mathrm{CRC32}(Q_2) \wedge \mathrm{CRC32}(Q_3))$

実際のホワイトニングは、以下のように実行されます:

128-bit ブロック暗号の場合:
T_1 = C の上位 64-bit ワード
T_2 = C の下位 64-bit ワード
$C' = (T_1 \wedge W) \,||\, (T_2 \wedge W)$

64-bit ブロック暗号の場合:
$C' = C \wedge W$

ヘッダーキーの生成、ソルト、反復回数

ヘッダーキーは TrueCrypt ボリュームヘッダーの暗号化領域を復号するのに使われます。(「暗号化の仕組み」と「TrueCrypt ボリュームフォーマット仕様」を参照)　TrueCrypt　ヘッダーキーを生成する手法は PBKDF2 であり、PKCS #5 v2.0 に規定されています。[7]を参照。

512-bitソルト(ボリューム作成プロセスで組み込みの乱数発生機構で生成されるランダム数)が使われます。ということは、それぞれのパスワードについて 2^{512}(2 の 512 乗)のキーがあるということです。これは、オフライン辞書攻撃に対する脆弱さを非常に大きく減少させます。(ソルトが使われると、事前にすべてのキーをコンピュータで組み合わせてパスワード辞書を作るということは、非常に難しくなります) [7]　ヘッダーキーを生成するにはキー生成関数を 2000 回繰り返さなくてはいけません。これは徹底したパスワード探索に要する時間を非常に増大させます。(brute force attack)[7]　ヘッダーキー生成関数は、HMAC-SHA-1 か HMAC-RIPEMD-160 に基づいており、ユーザーはどちらかを選択できます。生成されるキーの質も長さも、基礎となるハッシュ関数の出力サイズに制限されません。(生成されるキーはつねに要求されたサイズになります。160 ビットとは限りません)
詳細は[7]を参照してください。

カスケードの個々の暗号が使うヘッダーキーは、同じパスワードから生成されますが、相互に独立しています。たとえば、AES-Twofish-Serpent では、ヘッダーキー生成関数はパスワードから 768-bit キーを生成するように指示を受けます。その後、このキーは三つの 256-bit キーに分割され、最初のものが Serpent で、二番目のものが Twofish、三番目のものが AES で使われます。

乱数発生機構

TrueCrypt に実装している乱数発生機構は、マスター暗号化キー、ソルト、および IV とホワイトニング値を作るのに使われる値を生成することに使われます。(「動作モード」と「ホワイトニング」を参照)

乱数発生機構は RAM(メモリ)にランダム値の集合(プール)を作ります。256 バイトの大きさの集合は以下をもとにしたデータで満たされます。

- TrueCrypt ボリューム作成ウィザードのウィンドー内でのマウスの動き(CRC32-hash で処理されたマウスの座標、イベントの間隔とシステム時刻):
  CRC32(*MouseCoordinates*) || CRC32(*EventDeltaTime* || *EventTime*)

- TrueCryptボリューム作成ウィザードのウィンドー内でのマウスのクリック(CRC32-hashで処理されたボタンID、イベントの間隔とシステム時刻):
  CRC32(*MouseButtonID*) || CRC32(*EventDeltaTime* || *EventTime*)

- マウスポインタが TrueCrypt ボリューム作成ウィザードのウィンドー内にあるときのキーストローク(CRC32-hash で処理されたキーコード、イベントの間隔とシステム時刻):
  CRC32(*KeyID*) || CRC32(*EventDeltaTime* || *AbsoluteEventTime*)

- ディスクドライブのパフォーマンス統計

- ネットワークインターフェース統計(NETAPI32)

- MS Windows 暗号 API

- さまざまな Win32 ハンドル、時間変数、カウンタ(250-ms ごとに収集)

マウスやキーストロークのイベントは、それぞれの最後と最後から二番目とは異なる場合のみ採用されます。*EventDeltaTime* は最後に採用したイベントと現在のものとの時間差を意味します。*AbsoluteEventTime* はイベントが採用されたシステム時刻です。

上記のものから得た値を集合に書き込む前に、値はバイトに分割されます。(CRC-32 の 32-bit出力は 4 バイトに分割されます)　これらのバイトはmodulo 2^8 addition処理をして、集合のカーソル位置に個々に(集合の古い値を置き換えるのではなく)書き込まれます。バイトが書き込まれたあと、カーソル位置は 1 バイト進みます。カーソルが集合の最後に到達すると、カーソルは集合の最初に戻ります。さらに、値(バイト)が集合に追加されたあと、集合はハッシュ関数(SHA-1 か RIPEMD-160 – どちらかユーザーが選択したほう)でハッシュされます。

乱数発生機構の設計と実装は以下に基づいています:

- *Software Generation of Practically Strong Random Numbers* by Peter Gutmann [10]
- *Cryptographic Random Numbers* by Carl Ellison [11]

TrueCrypt ボリュームフォーマット仕様

TrueCrypt ボリュームには署名のようなものはありません。復号されるまでは、すべてがランダムなデータにしか見えません。したがって、TrueCrypt コンテナやパーティションであるかどうかを判断することはできません。

それぞれの TrueCrypt ボリュームはボリュームが作られるときにランダム値で満たされます。(クイックフォーマット時を除く)　ランダム値は以下のように生成されます: TrueCrypt ボリュームのフォーマットが始まる直前に臨時の暗号化キー、プレーンテキストブロック、IV、ホワイトニング・シードが組み込みの乱数発生機構で生成されます。(これらすべては RAM に保持され、フォーマットが完了すると廃棄されます)　ユーザーが選んだ暗号化アルゴリズムは臨時キーで初期化され、それが作り出した暗号テキストブロックボリュームの空き領域を満たす(上書きする)のに使われます。IV は通常どおり生成(「動作モード」参照)されますが、IV シードはボリュームから取得できないので、乱数発生機構から生成されます。ホワントニングも通常どおり(「ホワイトニング」参照)適用されますが、ホワイトニング値は乱数発生機構で生成される値から導かれます。

TrueCrypt ボリュームフォーマット バージョン 1　仕様

オフセット (bytes)	サイズ (bytes)	暗号化	備考
0	64	非暗号化	ソルト*
64	4	暗号化	ASCII 文字列 “TRUE”
68	2	暗号化	ボリュームフォーマットバージョン
70	2	暗号化	ボリュームを開く最小プログラムバージョン
72	4	暗号化	(復号) 256-511 バイトの CRC-32 チェックサム
76	8	暗号化	ボリューム作成日時
84	8	暗号化	ヘッダー作成/変更日時
92	8	暗号化	予約(0 をセット)
100	156	暗号化	未使用
256	32	暗号化	IV とホワイトニング値を生成するためのデータ
288	224	暗号化	マスター暗号化キー†
512	*N/A*	暗号化	データ領域(実際のボリュームの内容)

TrueCrypt ボリュームが(空き領域に)隠しボリュームを持つ場合には、隠しボリュームのヘッダーは主ボリュームの最後から 1536 のオフセットになります。(主/外殻ボリュームのヘッダーはボリュームの最初にあります。-「隠しボリューム」を参照)　隠しボリュームのフォーマットは下表のとおりです。

オフセット (bytes)	サイズ (bytes)	暗号化	備考
0	64	非暗号化	ソルト
64	4	暗号化	ASCII 文字列 “TRUE”
68	2	暗号化	ボリュームフォーマットバージョン
70	2	暗号化	ボリュームを開く最小プログラムバージョン
72	4	暗号化	(復号) 256-511 バイトの CRC-32 チェックサム
76	8	暗号化	ボリューム作成日時
84	8	暗号化	ヘッダー作成/変更日時
92	8	暗号化	隠しボリュームのサイズ
100	156	暗号化	未使用
256	32	暗号化	IV とホワイトニング値を生成するためのデータ
288	224	暗号化	マスター暗号化キー‡

* ソルトは暗号化する必要がありません。秘密にする必要がないからです。[7](ソルトは一連のランダム値です)
†ボリュームが暗号のカスケードで暗号化されている場合には、マスターキーは複数になります。
‡ボリュームが暗号のカスケードで暗号化されている場合には、マスターキーは複数になります。

0-63 バイト(ソルト)、256-287 バイト(IV とホワイトニング値生成に使うデータ)、288-511 バイト(マスター暗号化キー) は組み込み乱数発生機構を使って生成されたランダム値が入ります。(「乱数発生機構」参照)

準拠規格

TrueCrypt は以下の規格、仕様、勧告に準拠しています。

- PKCS #5 v2.0 [7]
- FIPS 46-3 [13]
- FIPS 197 [3]
- FIPS 198 [22]
- FIPS 180-2 [14]
- NIST S. P. 800-38A [12]

ソースコード

TrueCrypt の完全なソースコード(C とアセンブラ)は次のところで入手できます:
http://truecrypt.sourceforge.net/downloads.php

今後の開発予定

将来の計画に含まれている機能については以下を参照してください:
http://truecrypt.sourceforge.net/future.php

連絡先

私たちへの連絡についての情報:
http://truecrypt.sourceforge.net/contact.php

バージョン履歴

3.1a

2005 年 2 月 7 日

バグ修正

- リムーバブルメディアとしてマウントされたボリュームも、(chkdsk.exe による)チェック/修復、デフラグ、フォーマットなどができるようにした。
- ボリューム作成ウィザードは、*ツール -> 設定*でセットされたデフォルトマウントオプションを反映するようにした。
- 特定のシステムでマウント/アンマウントが失敗するバグを修正。
- TrueCrypt アンインストーラはインストール中にかならずインストールされるようにした。
- (コマンドラインの) */volume* オプションに相対パスが使えるようにした。
- 「すべてをアンマウント」したあとに、Windows Explorer(マイコンピュータ)でドライブ A:が消えることがなくなった。
- その他小さいバグの修正。

機能改善

- トラベラーモードで動作している場合は、TrueCrypt ドライバーは不要になればアンロードされる。 (メインアプリケーションとボリューム作成ウィザードが終了し、TrueCrypt ボリュームがまったくマウントされていない場合)
- 読み取り専用か読み書き可かというアクセスモードをボリュームプロパティダイアログに表示されるようにした。
- その他の小さい改善。

3.1

2005 年 1 月 22 日

機能改善

- 他のドライバー(通常はアンチウィルス)がすでに使用中であるパーティション/デバイスを、マウントすることができるようにした。
- 複数のボリューム作成ウィザードを走らせることができるようにした。

新機能

- TrueCrypt をトラベラー(旅行者)モードで動かすことができるようにした。これで、TrueCrypt を稼働する OS に対してインストールしなくてもいいということになる。トラベラーモードで TrueCrypt を起動するのは以下の二通り:

1) バイナリ配布パッケージを展開し、(インストールせずに)直接 TrueCrypt.exe を走らせる。

2) ツールメニューにある「トラベラーディスク作成」で特別なトラベラーディスクを作り、そこから TrueCrypt を起動する。この機能は、トラベラーディスクが挿入されたとき、自動的に特定のボリュームがマウントされるように設定することも可。(これはトラベラーディスクが CD や DVD のようなリムーバブルメディアであるときだけに有効。USB メモリスティックの場合には Windows XP SP2 が必要)

- ボリュームを読み取り専用でマウントできるようにした。これは新しい「マウントオプション」ダイアログにあり、マウント時のパスワード入力ダイアログから開くことができる。(コマンドラインでは: /mountoption ro)

- ボリュームをリムーバブルメディアとしてマウントできるようにした。(たとえば、ボリュームに Windows が *Recycled* や *System Volume Information* フォルダーを作ることを防止するために) これは新しい「マウントオプション」ダイアログにあり、マウント時のパスワード入力ダイアログから開くことができる。(コマンドラインでは: /mountoption rm)

- メインプログラムの設定でデフォルトマウントオプションを設定できるようにした。

- 「ドライブリストの更新」機能をツールメニューに追加した。これは Windows Explorer が新規マウントされたボリュームを登録するのに失敗した場合に使える。(新しいドライブがマイコンピュータに表示されない場合)

- ボリュームを TrueCrypt プログラム・ウィンドーにドラッグすることで「選択」をすることができるようにした。(同時に、Windows のファイルセレクタを避ける)

- '/auto device'(コマンドプロンプト)ですべてのデバイス/パーティション型 TrueCrypt ボリュームを自動マウントできるようにした。

バグ修正

- デバイスの自動マウント機能で、ある種の(USB メモリスティックのような)リムーバブルメディアにできる幽霊パーティションをマウントしないようにした。

- ある種の(USB メモリスティックのような)リムーバブルメディアで TrueCrypt がすべての使用可能な空き領域を使わないことがあった。
  補足: このバグは E4M に起因するもので、E4M で作成されたボリュームにかかわるものである。

- 警告: USB メモリスティックのようなリムーバブルメディア上には TrueCrypt 3.0 または 3.0a で作られた隠しボリューム(ファイル形式コンテナは除く)をマウントできないということである。なぜなら隠しボリュームの想定される位置はホストボリュームの大きさによって変わるためである。そのような場合には、TrueCrypt 3.1 にアップグレードする前に、リムーバブルではないメディア上の臨時 TrueCrypt ボリュームまたはリムーバブルメディアの「隠し」ではないボリュームにすべてのファイルを退避し、古い隠しボリュームのデータを臨時ボリュームへ移動すること。その後に TrueCrypt 3.1 をインストールし、隠しボリュームを作成し、臨時ボリュームからそこへファイルを移動する。

- TrueCrypt ボリュームのマウント/アンマウント時に、ドライブ変更のメッセージに対し無応答となりフリーズするということは、発生しなくなった。

- FAT ボリュームに小さすぎるクラスタを設定できないようにした。(これはさまざまな問題を起こす)

- コマンドラインパーサが TrueCrypt のクラッシュを起こさないようにした。
- その他の小さいバグ修正

3.0a

2004 年 12 月 11 日

バグ修正

- TrueCrypt ボリュームがマウントされているときに、他の Twofish か Serpent で暗号化されているボリュームに書き込みをしても、データが破損しないようにした。(Twofish か Serpent を使った暗号カスケードの場合も同様)
- その他の小さいバグ修正

3.0

2004 年 12 月 10 日

新機能

- 隠しボリューム(ファイルコンテナ、パーティション/デバイス)の作成、マウント機能。これは、ユーザーが敵対者にパスワードを明かすよう強要され拒否できない場合(たとえば相手が暴力に訴えるような場合)に、問題を解決する。

  他の TrueCrypt ボリュームの空き領域に TrueCrypt ボリュームを作るというのが、ポイントである。外殻ボリュームがマウントされるとき、それが隠しボリュームを含むかどうかを判断することはできない。なぜなら、どの TrueCrypt ボリュームの空き領域も作成時にランダム値で埋められている(クイックフォーマット時を除く)からであり、隠しボリュームのどの部分もランダムデータと区別できないからである。

  隠しボリュームのパスワードは、外殻ボリュームのパスワードとは異なったものでなくてはいけない。隠しボリュームを作成する前に、外殻ボリュームには本当には隠そうとは思っていない何か秘密情報らしいファイルをいくつかコピーしておく。これらのファイルは、パスワードを明かすことを強要する人に見せるためのものである。隠しボリュームのパスワードは守り、外殻ボリュームのものだけを明かせばよい。本当に秘密にしたいファイルは隠しボリュームに入れること。

  隠しボリュームが外殻ボリュームのデータを上書きしてしまわないように隠しボリュームの大きさを決めるのは初心者には非常に難しいので、ボリューム作成ウィザードは隠しボリューム作成の前に自動で外殻ボリュームのクラスタ配置をスキャンし隠しボリュームの作成可能な最大サイズを決定する。

- Serpent 暗号化アルゴリズム(256-bit キー)
- Twofish 暗号化アルゴリズム(256-bit キー)
- 強制アンマウント(システムか何かのアプリケーションがそのボリュームのファイルを使っている場合でも、アンマウント可能)
- 暗号カスケードを追加(AES-Twofish-Serpent, AES-Blowfish など) カスケードのそれぞれの暗号は各自の暗号化キーを持つ。(キーは相互に独立している)

- システムか何かのアプリケーションがそのボリュームを使っている場合でも、マウント可能にした。(共有アクセスモード)
- システムか何かのアプリケーションがデバイス/パーティションを使っている場合でも、暗号化可能にした。
- 「デバイスの選択」ダイアログとパーティションの自動マウント機能で、パーティションがないデバイスをサポートすることとした。
- ツールメニューとボリューム作成ウィザードに、暗号化アルゴリズムベンチマーク機能を追加した。
- ボリュームの新規作成とパスワード変更時に、CapsLock が On だと警告を出すこととした。
- コマンドラインで、/l がなく/a が指定されているときには、最初の空きドライブレターを使うこととした。
- コマンドラインオプションの追加: /force または/f　強制アンマウントと共有モード(排他なしのアクセス)のマウント。
- ドライブ文字を「デバイスの選択」ウィンドーに表示するようにした。

バグ修正

- ボリュームをアンマウントするときに、ブルースクリーン(システムクラッシュ)が起きないようにした。(このバグは E4M に起因する)
- パーティションがシステムかアプリケーションに使用されていても「デバイスの選択」ダイアログが表示されるようにした。
- パーティション/デバイスの大きさが 1024 バイトの倍数ではない場合に、最後のセクター(512 バイト)が TrueCrypt ボリュームに使われなかった。(ボリュームがパーティション/デバイスより 512 バイト小さかった)
  補足: このバグは E4M に起因する。したがって、この現象は E4M で暗号化されたパーティション/デバイスで発生する)
- サイズが正確に 129MB のボリュームの FAT ボリュームでも空き領域がゼロにならない。(前のバージョンで作成した 129MB FAT ボリュームでは空き領域がゼロになる)
- 管理者権限がないユーザーでも Windows Server 2003 でならファイルコンテナが作成できるようにした。
- その他の小さいバグ修正。

機能改善

- コンテナのタイムスタンプ(コンテナの最終アクセス、最終更新日時)は TrueCrypt がアクセスすることでは更新されないようにした。(アンマウント、マウント試行、パスワードの変更または変更の試行、およびその中への隠しボリュームの作成)
- TrueCrypt サービスは不要にし、削除した。この機能は TrueCrypt ドライバーが扱う。
- 「履歴を保存しない」にチェックを入れた場合に、Windows がファイルセレクタ履歴や「最近使ったフ

ァイル」に最後にアクセスしたファイルコンテナのファイル名を保存しないようにした。

- その他小さい改善。

その他

- TrueCrypt は Windows “Longhorn”(Windows XP の次バージョンのベータ版)での動作テストが成功した。

- ユーザーがパスワードの最短文字長に制限されないようにした。(警告が表示され、確認を求める)

2.1a

2004 年 10 月 1 日

機能削除

- IDEA 暗号化アルゴリズムを削除。これで非営利/営利組織が TrueCrypt を使うために、個別に IDEA ライセンスを取得する必要がなくなった。(IDEA ライセンスに従えば、非営利/営利組織による IDEA アルゴリズムを含むソフトウェアの使用は、すべて商業目的の利用とみなされ、MediaCrypt AG のライセンスに従うことになる)

  重要: IDEA 暗号化アルゴリズムで暗号化された TrueCrypt ボリュームは TrueCrypt 2.1a ではマウントできない。このようなボリュームがあるならば、TrueCrypt 2.1a にアップグレードする前に、IDEA 以外で作成した TrueCrypt ボリュームを作り、そちらへファイルを移動しておくこと。

2.1

2004 年 6 月 21 日

新機能

- RIPEMD-160 ハッシュ・アルゴリズムを追加。ユーザーは TrueCrypt でどちら(SHA-1 か RIPEMD-160)のハッシュ・アルゴリズムを使うかを選択できる。

  補足: RIPEMD-160 はオープン・アカデミック・コミュニティで設計され、NASA と NIST で設計された SHA-1 と同等の代替品である。前バージョンでは SHA-1 の大きな脆弱さのためにプログラム全体が実質的に使い物にならなくなる危険性があった。ユーザーが選択したハッシュ・アルゴリズムは新規ボリューム作成時にランダム値を生成することと、ヘッダーキー生成関数に使われる。(PKCS #5 v2.0 で明記されているように、ハッシュ関数に基づいた HMAC) 乱数発生機構は、マスター暗号化キー、ソルト、IV とホワイトニング値生成に使われる。

- ボリュームパスワードを変更するときに、新しいボリュームヘッダーキーを生成するのに使う HMAC ハッシュ・アルゴリズムを選択できるようにした。

- NTFS TrueCrypt ボリュームおよび未フォーマット TrueCrypt ボリュームを作成できるようにした。この拡張によりボリュームサイズを 2048GB に制限しなくなった。(TrueCrypt の前バージョンでは FAT ボリュームのみが作成可能であった。暗号化されていてもいなくても、FAT ボリュームのサイズは 2048GB を越えることはできない)

- ヘッダーキーの内容を(ソルト表示に代えて)ボリューム作成ウィザードのウィンドーに表示する。
- ランダム値プール、マスターキー、ヘッダーキーの内容をボリューム作成ウィザードのウィンドーに表示しないようにできるようにした。

バグ修正

- 他の TrueCrypt コンテナに格納された TrueCrypt コンテナがマウントされているときに、「すべてアンマウント」機能を使ってどちらもアンマウントできるようにし、ブルースクリーンでシテスムシャットダウンということは発生しない。
- コマンドラインの扱いについての小さなバグ修正。

機能改善

- ドライバーについてのいくつかの小さな改善。

その他

- GPL ライセンスに関わる問題を避けるため、オリジナルの E4M ライセンスのもとにリリースした。(IDEA 特許情報と特定の法的な注意を追加)

2.0

2004 年 6 月 7 日

バグ修正

- TrueCrypt パーティションが重い並行処理(TrueCrypt パーテシィョンから、またはそこへファイルをコピーするときなど)をしているときにデータが破損しないようにした。TrueCrypt パーティションにあるファイルにアクセスできないことがあるという問題も解決した。

  補足: ファイル形式ボリュームでは、この問題の影響はない。
- ボリュームをアンマウントし再マウントしたあと、ファイルシステムが OS に正しく認識され、ドライブレターを再使用できるようにした。(Windows 2000 関連)
- コマンドラインでクワィエットモードを指定したときには、メインプログラム・ウィンドーは表示しないようにした。
- コマンドラインで、ボリュームをマウントするために二つのパスワード入力を不要にした。
- パーティションのどれかが OS からアクセスできなくても、すべてのパーティションを TrueCrypt で見えるようにした。(これまではアクセスできないパーティションがあると、すべてのパーティションが見えなくなっていた)
- コマンドラインでファイル形式ボリュームをマウントするときに、相対パス指定を可能とした。
- コマンドラインで自動マウントをするときに、不正なパスワードを報告することにした。

新機能

- AES-256(Rijndael)暗号化アルゴリズム
- コマンドラインオプションの /dismountall を /dismount とリネームし、ドライブレターを指定すれば特定の一つのボリュームをアンマウントできるようにした。

機能改善

- TrueCryptボリュームの暗号化キーやホワイトニング・シードを含むメモリ領域をロックし、Windowsページファイルにスワップされないようにした。
- ランダムプールの状態が直接は利用されないようにした。これによってプールの内容が漏れることがなくなる。

その他

- GNU General Public License (GPL)にしたがってリリースした。

1.0a *(by TrueCrypt Team)*

2004 年 2 月 3 日

機能削除

- TrueCrypt は Windows 98/ME では動作しなくした。

1.0 *(by TrueCrypt Team)*

2004 年 2 月 2 日

TrueCryptはE4M*(Encryption for the Masses)*に基づいている。したがって以下のリストはTrueCryptとE4M 2.02aの差異についての記事を含む。(小さな差異は除く)

機能改善

- Windows XP/2000をサポート
- 最大ボリュームサイズは18,446,744,073 GB (E4M は2 GBまで).
  補足: 最大ボリュームサイズを決定するには、ファイルシステム、ハードウェア接続規格、OSの制限などを考慮すること。
- 「もっともらしい否認」TrueCryptコンテナやパーティションを特定することはできない。復号されるまでは、TrueCryptボリュームはランダムデータの集まりにしか見えない。(署名のようなものもつかない) したがってファイル、パーティション、デバイスがTrueCryptボリュームであるとか、暗号化されているとかを証明することはできない。「もっともらしい否認」をするには、ボリュームのフォーマットと暗号化プロセスを大きく変更する必要があった。

- ソルトを 64 バイトとした。(E4M は 20 バイト)

- キー生成関数の反復回数を 2,000 回に増やした。(E4M は 1,000 回)

- ボリューム作成時に、空き領域すべてにゼロを書き込む代わりに、ランダムデータを書き込むこととした。これでプレーンテキストの予測を減らし、将来的には隠しボリュームについての「もっともらしい否認」のレベルをあげることになる。

- 1 台のディスクあたり 32 パーティションまで暗号化できる。(Windows XP/2000)

- 最小のボリュームパスワードの長さを 12 文字までに増やした。

- 最大ボリュームパスワードの長さを 100 から 64 に減らした。これは以下のことを避けるためである: 64 文字を越えるパスワードを HMAC-SHA-1 へ渡すと、パスワード全体が SHA-1 でハッシュされ、その結果の 160-bit 値がオリジナルパスワードの代わりに使われる。(HMAC-SHA-1 の仕様による) ということで、パスワードの長さは実質的に短くなってしまう。

- Blowfish キーの長さを 448 ビットに増やした。
  補足: キーサイズを 448 ビットに増やしたのはやりすぎのように見えるかもしれないが、これには特別な理由はない。(暗号化/復号の速度に影響はない)

バグ修正

- セクター・スクランブル・アルゴリズムの欠陥を修正。E4M で暗号化後、暗号化しようとする同じ値からなる(全部ゼロの場合など)二つ以上のセクターが同じ 8 バイト値の繰り返しになる。(これらの暗号化されたセクターの最初の 8 バイトが他の暗号化されたセクターの最初の 8 バイトと同じになってしまう) これが修正されないと、「もっもとらしい否認」が不可能になることもあった。

- TrueCrypt ボリュームをアンマウントできるようにした。(Windows XP 関連)

- 一つ以上の TrueCrypt ボリュームがマウントされているときに、Windows をシャットダウンするとブルースクリーンになることはなくなった。

- ドライブのジオメトリを正しく計算するようにした。(chkdsk.exe と format.exe が失敗することはなくなった)

- Windows 標準のフォーマットツールで TrueCrypt ボリュームを FAT32 または NTFS で再フォーマットできるようにした(Windows XP/2000 関連)

- Windows のチェックディスクを TrueCrypt ボリュームでも使えるようになった。(Windows XP/2000 関連)

- Windows のデフラグを TrueCrypt ボリュームでも使えるようになった。(Windows XP/2000 関連)

新機能

- 新しい IV(Initial Vector)生成アルゴリズム(詳細はドキュメント参照)

- 各セクターの 8 バイトずつは(セクターが暗号化されたあとで)ランダムな 64 ビット値と XOR される。この 64 ビット値はそれぞれのセクター(512 バイト長)とボリュームでユニーク(他と同一ではない)である。これは、プレーンテキストと暗号化テキストのペアを得るのを少し難しくする。

- ボリューム履歴を削除する新機能
- パーティション/デバイスを選択するとき、選択できるパーティション/デバイスのサイズとファイルシステムの種類を表示する。(Windows XP/2000)
- マウントされた TrueCrypt ボリュームのリストに、サイズと使われた暗号化アルゴリズムを表示する。(Windows XP/2000)
- 空き領域の大きさを表示する(マイコンピュータ他)
- Windows XP のフォーマット機能は FAT32 では 32GB を越えるボリュームをサポートしていない。しかし TrueCrypt ボリューム作成ウィザードは 32GB を越える FAT32 ボリュームを作成できる。
- 正しいパスワードが入力されれば(キャッシュにパスワードがあればそれでもいい)複数の TrueCrypt パーティションをマウントできる新機能
- クイックフォーマット(パーティション/デバイスのみ)
- クラスタサイズの選択(新ボリューム作成時)
- ボリュームプロパティを見ることができる。(暗号化アルゴリズム、ボリューム作成日時、最終パスワード変更日時 など)
- TrueCrypt ボリュームをすべてアンマウントする新機能
- マウントされたすべての TrueCrypt ボリュームをアンマウントするコマンドラインの新オプション: /d と /dismountall
- HMAC-SHA1 と CRC-32 アルゴリズムテストを自己診断機能に含めた。
- プログラムメニューと設定ウィンドーを追加した。
- ユーザーインターフェースのフォントのカスタマイズ
- 任意だが、TrueCrypt インストーラがシステム復元ポイントを作成できるようになった。(Windows XP/ME)
- パスワード入力フィールドは正しいボリュームパスワードが入力されたあとクリアされる。
- 新しいグラフィックス、アイコン、ユーザーインターフェース
- 新しいドキュメント

機能削除

- E4M と SFS ボリュームのサポート停止
- DES 暗号の除去
- HMAC-MD5 の除去(HMAC-RIPEMD-160 に置換え予定)

謝辞

私たちは以下のみなさんに感謝します:

Paul Le Roux 彼の E4M ソースコードを入手できるようにしてくれました; TrueCrypt は E4M に基づいています。

Eric Young すばらしい libdes, libcast などを書いてくれました。これらは TrueCrypt の暗号の一部のソースです。

Dr. Brian Gladman 彼はすばらしい AES, Serpent, and Twofish ルーチンを書いてくれました。

Peter Gutmann 彼の乱数についての論文と、TrueCrypt の乱数発生機構の一部のソースである cryptlib を作ってくれたこと。

Andy Neville ファイル形式のボリューム(E4M)を実装するのに役立ったコードと提案について。

David Kelvin プライバシー・パスワードとクワイエットモードのコマンドライン引数を付け加えてくれました。

暗号化とハッシュ・アルゴリズムの設計者のみなさん:
Horst Feistel, Don Coppersmith, Whitfield Diffie, Martin Hellman, Walt Tuchmann, Lars Knudsen, Ross Anderson, Eli Biham, Bruce Schneier, David Wagner, John Kelsey, Niels Ferguson, Doug Whiting, Chris Hall, Joan Daemen, Vincent Rijmen, Carlisle Adams, Stafford Tavares, Hans Dobbertin, Antoon Bosselaers, and Bart Preneel.
このプロジェクトを可能にしてくれたみなさん、精神的に支援してくれたみなさん

ありがとうございました。

参考文献

[1] C. Adams, *Symmetric cryptographic system for data encryption*, U.S. Patent 5,511,123, filed August 4, 1994, issued April 23, 1996, available at http://patft.uspto.gov/.

[2] U.S. Committee on National Security Systems (CNSS), *National Policy on the Use of the Advanced Encryption Standard (AES) to Protect National Security Systems and National Security Information*, CNSS Policy No. 15, Fact Sheet No. 1, June 2003, available at http://www.nstissc.gov/Assets/pdf/fact%20sheet.pdf.

[3] NIST, *Advanced Encryption Standard (AES)*, Federal Information Processing Standards Publication 197, November 26, 2001, available at http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.

[4] J. Nechvatal, E. Barker, L. Bassham, W. Burr, M. Dworkin, J. Foti, E. Roback, NIST, *Report on the Development of the Advanced Encryption Standard (AES)*, October 2, 2000, available at http://csrc.nist.gov/CryptoToolkit/aes/round2/r2report.pdf.

[5] B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, N. Ferguson, T. Kohno, M. Stay, *The Twofish Team's Final Comments on AES Selection*, May 15, 2000, available at http://csrc.nist.gov/CryptoToolkit/aes/round2/comments/20000515-bschneier.pdf.

[6] C. Adams, *The CAST-128 Encryption Algorithm*, Request for Comments 2144, May 1997, available at http://www.rfc-editor.org/rfc/rfc2144.txt.

[7] RSA Laboratories, *PKCS #5 v2.0: Password-Based Cryptography Standard*, RSA Data Security, Inc. Public-Key Cryptography Standards (PKCS), March 25, 1999, available at ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-5v2/pkcs5v2-0.pdf .

[8] H. Krawczyk, IBM, M. Bellare, UCSD, R. Canetti, IBM, *HMAC: Keyed-Hashing for Message Authentication*, Request for Comments 2104, February 1997, available at http://www.rfc-editor.org/rfc/rfc2104.txt.

[9] P. Cheng, IBM, R. Glenn, NIST, *Test Cases for HMAC-MD5 and HMAC-SHA-1*, Request for Comments 2202, February 1997, available at http://www.rfc-editor.org/rfc/rfc2202.txt.

[10] Peter Gutmann, *Software Generation of Practically Strong Random Numbers*, presented at the 1998 Usenix Security Symposium, available at http://www.cs.auckland.ac.nz/~pgut001/pubs/usenix98.pdf.

[11] Carl Ellison, *Cryptographic Random Numbers*, originally an appendix to the P1363 standard, available at http://world.std.com/~cme/P1363/ranno.html.

[12] Morris Dworkin, *Recommendation for Block Cipher Modes of Operation*, NIST Special Publication 800-38A, 2001 Edition, available at http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf.

[13] NIST, *Data Encryption Standard*, Federal Information Processing Standards Publication 46-3, October 25, 1999, available at http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf.

[14] NIST, *Secure Hash Standard*, August 1, 2002, available at http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf.

[15] U. Maurer, J. Massey, *Cascade Ciphers: The Importance of Being First*, Journal of Cryptology, v. 6, n. 1, 1993

[16] Bruce Schneier, *Applied Cryptography*, Second Edition, John Wiley & Sons, 1996

[17] List of the approved cryptographic algorithms for the protection of Protected Information within the Government of Canada: http://www.cse-cst.gc.ca/en/services/crypto_services/crypto_algorithms.html.

[18] Serpent home page: http://www.cl.cam.ac.uk/~rja14/serpent.html.

[19] M. E. Smid, *AES Issues*, AES Round 2 Comments, May 22, 2000, available at http://csrc.nist.gov/CryptoToolkit/aes/round2/comments/20000523-msmid-2.pdf.

[20] Menezes, P. van Oorschot, S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, October 1996

[21] Peter Gutmann, *Secure Deletion of Data from Magnetic and Solid-State Memory*, first published in the Sixth USENIX Security Symposium Proceedings, San Jose, California, July 22-25, 1996, available at http://www.cs.auckland.ac.nz/~pgut001/pubs/secure_del.html

[22] NIST, *The Keyed-Hash Message Authentication Code (HMAC)*, Federal Information Processing Standards Publication 198, March 6, 2002, available at http://csrc.nist.gov/publications/fips/fips198/fips-198a.pdf.

[23] J. Kelsey, *Twofish Technical Report #7: Key Separation in Twofish*, AES Round 2 public comment, April 7, 2000