

マル秘分散マニュアル

Version 1.0.0



目次

製品概要	2
ファイルの暗号化・復号化とは.....	2
秘密分散とは.....	2
マル秘分散 Web 版.....	3
本ソフトウェアが稼働するコンピュータ(OS)	4
インストール方法	5
ライセンスについて	6
<クライアントコンピュータにて本ソフトウェアを使用される場合>	6
<サーバーコンピュータにて本ソフトウェアを使用される場合>	7
本ソフトウェアの使用方法	8
暗号化-1 (本ソフトウェアの起動)	8
暗号化-2 (メイン画面)	9
暗号化-3 (処理終了後)	11
復号化-1 (本ソフトウェアの起動)	13
復号化-2 (メイン画面)	13
復号化-3 (処理終了後)	14
ライセンスの入力.....	15
その他の機能.....	16
よくある質問.....	17
制約事項	19
コマンドモードの使用方法 (上級者・開発者向け).....	20
コマンドモードについて、はじめに	20
コマンドモードのご使用方法	20
暗号化する.....	21
復号化する.....	22
秘密分散(分散する)	23
秘密分散(復元する)	24
Base64 エンコード&デコード.....	25
ファイルのハッシュ値参照	26
ファイルの強力削除.....	27
ライセンスキーの入力.....	28
ライセンス情報の参照	28
サンプルアプリケーション	29

製品概要

マル秘分散は、ファイルの暗号化・復号化と、秘密分散のできるソフトウェアです。

ファイルの暗号化・復号化とは

パスワード(鍵)を知らない人でないと、ファイルを読めなくする処理です。

復号化はその逆で、パスワード(鍵)を知っている人が、暗号化されたファイルを元に戻す処理です。

暗号化にはさまざまなアルゴリズムが存在しますが、本ソフトウェアでは、OpenSSL (<http://www.openssl.org/>) を、コマンドモードで実行した暗号化時と同じ結果の得られるAESとCamellia(それぞれ、256bit,192bit,128bit の3つの鍵長)に対応しています。

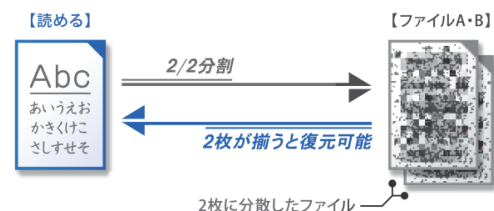
(図解)



秘密分散とは

1枚のファイルを複数枚のファイルに分散し、その複数枚のファイルがそろわないと、元のファイルに復元できなくする処理です(RSA 暗号の開発で世界的に有名なシャミア(Shamir)氏が、1979年に発表した論文の理論を元にした技術です)。

(図解: 2/2 分散)



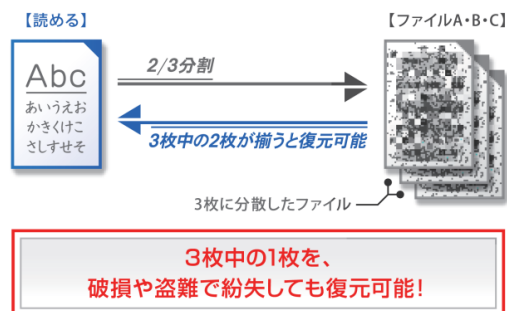
**ファイルAとファイルBを
別々の場所に保管すると安全!!**

分散された複数のファイルを、別々の箇所に保管しておけば、データ漏洩の可能性を劇的に下げることができます。

本ソフトウェアは、下記の2つに対応しています。

- ・ 2/2 分散(1ファイルを2ファイルに分散し、復元にはその2枚が両方とも必要)
- ・ 2/3 分散(1ファイルを3ファイルに分散し、復元にはそのうちの2枚が必要)

(図解:2/3 分散)



マル秘分散 Web 版

下記の URL から、マル秘分散の Web 版にアクセスすることができます。

<http://www.slogical.co.jp/product/hibun/>

マル秘分散 Web 版では、本ソフトウェアの一部の機能を、Web ブラウザから利用することが可能です (https で暗号化されたインターネット経由で、ファイルを送信し、暗復号化・秘密分散を行えます)。

例えば、本ソフトウェアで暗号化したファイルを第三者に送信する必要がある場合に、そのファイル受領者が、マル秘分散 Web 版で復号化するといったご利用方法も可能です。

本ソフトウェアが稼働するコンピュータ(OS)

本ソフトウェアには、Windows 版と Linux 版があります。

本マニュアルは Windows 版を想定して記載しておりますが、コマンドモードの仕様は Windows 版と Linux 版とで共通となります(Linux 版にはフォーム画面(GUI)が含まれず、サーバー処理やバッチ処理を想定したコマンドモード用プログラムのみが提供されます)。

システム要件の詳細は下記をご参照ください。

<Windows 版>

- ・ 本ソフトウェアは、Windows 2000, Windows XP, Windows Vista, Windows 2003, Windows 2008 用です(最新のセキュリティアップデートが適用されていることを想定しています)。
- ・ 本ソフトウェアを作動させるには .Net Framework Version 2.0 が必要となります。
- ・ 本ソフトウェアのマニュアルを参照するには、Adobe PDF 形式のファイルを表示可能なリーダーが必要となります。

<Linux 版>

- ・ 本ソフトウェアは、Red Hat Enterprise Linux v.4 用です。(それ以外のディストリビューションにも順次対応中です。お急ぎの場合にはお問い合わせください。)
- ・ 本ソフトウェアのマニュアルを参照するには、Adobe PDF 形式のファイルを表示可能なリーダーが必要となります。

Windows 版・Linux 版ともに、ハードウェアスペックとしては、Intel Pentium と互換性のある 500MHz 以上の CPU と、メモリ 512M 以上を搭載した IBM PC/AT 互換機を推奨しておりますが、お客様の性能要件に見合うかどうかは本ソフトウェアの試用期間中にご確認ください。

※Windows は米国 Microsoft Corporation の、米国およびその他の国における登録商標または商標です。

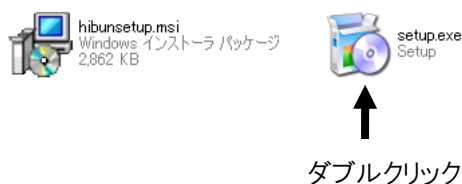
※.NET Framework は、米国 Microsoft Corporation の、米国およびその他の国における登録商標または商標です。

※その他すべての製品名および会社名は、各社の商標、または登録商標です。

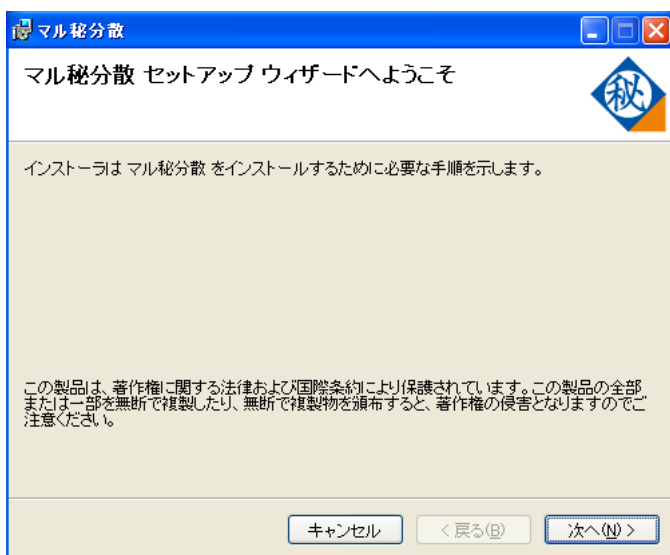
インストール方法

※重要: 本ソフトウェアのインストールは、管理者権限のあるユーザーで行ってください。

インストーラのフォルダに含まれる setup.exe をダブルクリックすることで、本ソフトウェアのインストールが開始されます。



setup.exe のダブルクリック後は、下図のような画面が表示されますので、画面の案内に従いインストールを行ってください。



インストールの終了後、30日間は、試用期間として本ソフトウェアをライセンスの購入無しにご利用いただくことが可能です。試用期間内に本ソフトウェアの動作確認をして頂けますようお願い致します。

※本ソフトウェアを起動させるには .Net Framework Version 2.0 が必要となりますが、コンピュータに .Net Framework がインストールされていない場合には .Net Framework のインストールも自動的に行われます。

※Linux 版は、提供される tar.gz ファイルを解凍することでインストールが完了します。

ライセンスについて

本ソフトウェアに関する権利などの詳細は「ソフトウェア使用許諾契約書」をご参照ください。

本ソフトウェアをご使用いただくために必要になるライセンスの必要購入数について、「ソフトウェア使用許諾契約書」より下記に抜粋して記載します。

<クライアントコンピュータにて本ソフトウェアを使用される場合>

- ・本ソフトウェアをインストールするクライアントコンピュータの数と、本ソフトウェアがインストールされたクライアントコンピュータを使用するユーザーの数のうち、大きい数量分のライセンスが必要になります。
- ・例えば、1人のユーザーが、本ソフトウェアを1台のクライアントコンピュータにてご使用になる場合は、1ライセンスの購入が必要になりますが、1人のユーザーが、本ソフトウェアを2台のクライアントコンピュータにてご使用になる場合には、2ライセンスの購入が必要になります。
- ・また、2人のユーザーが、本ソフトウェアがインストールされた1台のクライアントコンピュータを共有で使用される場合にも、2ライセンスの購入が必要になります。
- ・ただし、お客様が法人ではなく個人の場合には、1ライセンスあたり、3台までのクライアントコンピュータにて本ソフトウェアを使用することが可能です。

＜サーバーコンピュータにて本ソフトウェアを使用される場合＞

- ・本ソフトウェアの機能の一部を利用可能なサーバー環境を構築し、お客様組織内の不特定数のユーザーがそれにアクセスする場合（本ソフトウェアのコマンドモードを、お客様の独自サーバープログラムから呼び出す場合など）には、上記クライアントコンピュータ向けのライセンスとは異なる、サーバーコンピュータ用ライセンスの購入が必要となります。
- ・本ソフトウェアをインストールするサーバーコンピュータ1台につき、サーバーコンピュータ用ライセンスを1つご購入いただく必要があります。
- ・サーバーコンピュータに搭載された CPU の数は、必要ライセンス数のカウントに影響を与えることはありません。
- ・ただし、OS 環境を仮想化して、1 台のサーバーコンピュータに複数の OS 環境を構築し、そのそれぞれの OS 環境に本ソフトウェアをインストールする場合には、OS 環境分のライセンス数が必要となります。
- ・サーバーコンピュータ用ライセンスは、組織（法人格を持つ会社など）に対して発行されます。本ソフトウェアの機能を利用可能なサーバー環境を、組織外の第三者に公開・再販することはできません。
- ・通常のサーバーコンピュータ用ライセンスでは、お客様の主催するオンラインサービスの会員向けに、本ソフトウェアの機能を提供することはできません。ただし、別途のライセンス契約によってそのような利用形態も可能となりますので、詳細はエスロジカルまでお問い合わせください。

お問い合わせ先メールアドレス： info@slogical.co.jp

本ソフトウェアの使用方法

暗号化-1（本ソフトウェアの起動）

下記の3種の方法で、本ソフトウェアを起動することができます。

起動方法-1

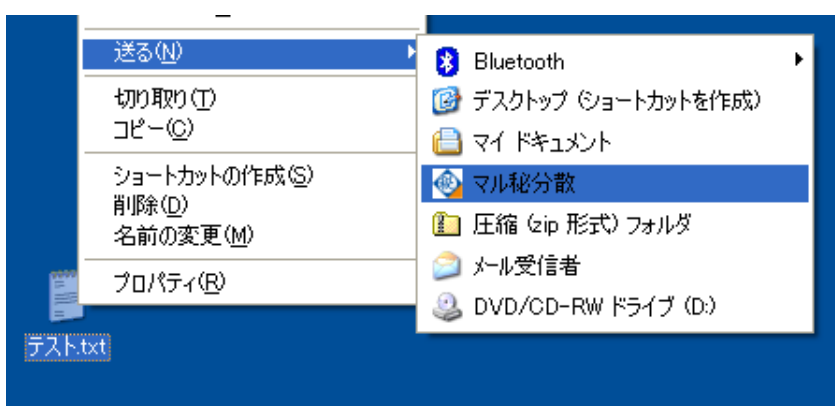
本ソフトウェアを起動するには、暗号化するファイル・フォルダを、デスクトップ上の「マル秘分散」アイコンにドラッグ&ドロップします。複数のファイルをまとめて暗号化する場合には、複数のファイルをまとめてマル秘分散のアイコンにドラッグ&ドロップしてください。

※「マル秘分散」アイコンは、本ソフトウェアのインストールによってデスクトップに自動生成されます。



起動方法-2

暗号化するファイル・フォルダを右クリックして、「送る」メニューから「マル秘分散」を選択することでも、本ソフトウェアを起動することができます。



起動方法-3

デスクトップ上の「マル秘分散」アイコンをダブルクリックして、本ソフトウェアを起動し、メイン画面のリストビューから処理対象ファイルを指定することも可能です（詳細は次ページをご覧ください）。

暗号化-2（メイン画面）

起動後、下図のような画面が現れますので、画面最下部の入力欄から、パスワードの入力後に実行ボタンをクリックします。

この窓に対して、ファイルをドラッグ&ドロップすることで、同時に複数のファイルを暗号化することが可能です。

フォルダをドラッグ&ドロップして、フォルダごと暗号化を行うことも可能です。

この窓内でファイルを右クリックして、暗号化の対象から除外することも可能です。

「暗号化」となっていることをご確認ください。

これらの動作オプションについては下記をご参照ください。

パスワードと、パスワード(確認)に同じパスワードを入力後、実行ボタンを押下します。

動作オプションの変更をせずに、AES256bitで暗号化を行うのみでしたら、ファイルの指定後にパスワードを入力し実行ボタンをクリックすることで暗号化が開始されます。

動作オプションとして、次ページのパラメータを変更することもできます。

◆暗号アルゴリズム

暗号アルゴリズム

AES256
 AES192
 AES128
 Camellia256
 Camellia192
 Camellia128
 Salted_ を付けない

AES と Camellia の2つの暗号化アルゴリズムに対して、256bit,192bit,128bitの3つの鍵長を選択することができます。

また、本ソフトウェアによる暗号化は、OpenSSL(<http://www.openssl.org/>)をコマンドラインで実行して暗号化したものと同等の結果が得られるように実装しておりますが、OpenSSL で暗号化を行うと、ファイルの先頭に「Salted_」というデータが追加される仕様となっています。

「Salted_ を付けない」というオプションをチェックすることで、この「Salted_」を出力しないようにすることができます。

◆秘密分散

秘密分散

なし 2/2分割 2/3分散

暗号化したファイルに対して、秘密分散処理を追加で行うかの指定を行うことができます。

◆base64

base64

なし あり

暗号化したファイルに対して、base64 エンコードを行うかの指定をすることができます。base64 すると、バイナリデータがテキスト化されますので、暗号化処理後のデータをテキストとして利用したい場合には、本オプションを「あり」と指定してください。

◆出力フォルダ

出力フォルダ (未指定時は、対象ファイルと同じフォルダに出力します)

1

2

3

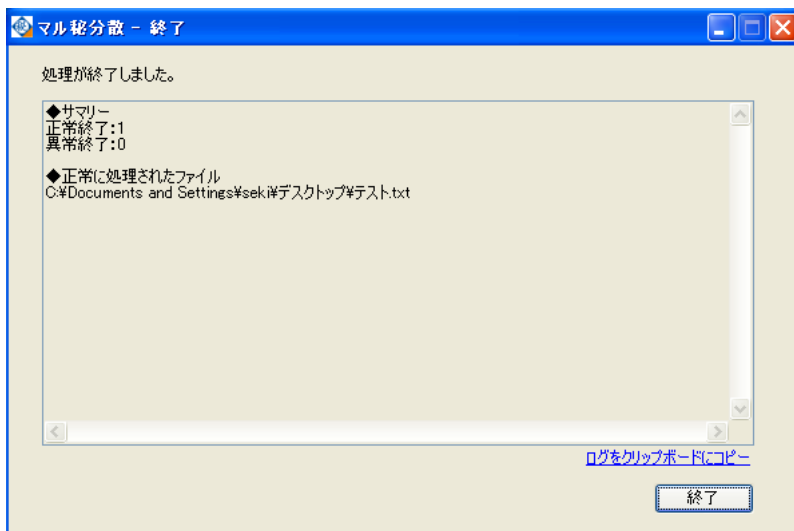
暗号化したファイルを出力するフォルダを指定することができます。

秘密分散時には、それぞれの分散ファイルごとに出力ディレクトリを指定することができますので、たとえば、2/2 分散で秘密分散を行い、1枚のファイルはコンピュータ内に保存し、もう1ファイルをUSBメモリ内に保存することなども可能となります。

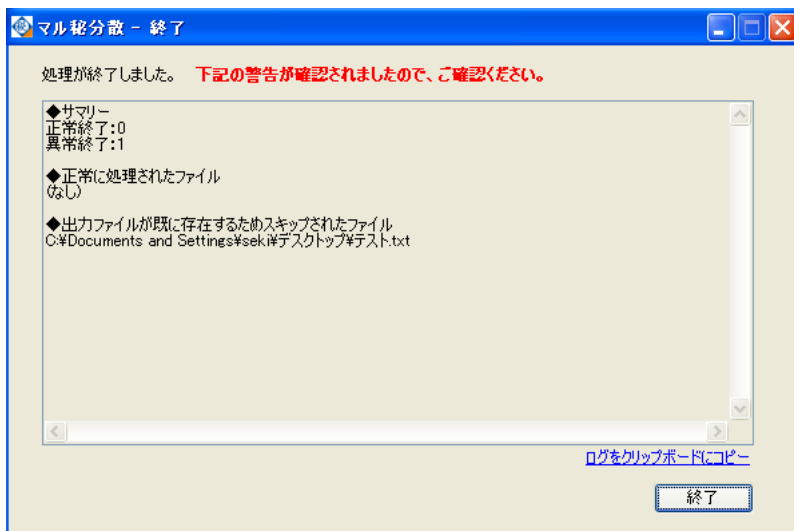
暗号化-3（処理終了後）

暗号化処理の終了後には、次のようなメッセージ画面が表示されます。

（正常に終了した場合）



（警告が確認された場合）



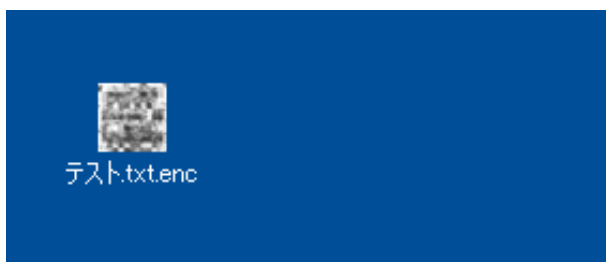
※警告がある場合には、上図のように赤文字でワーニングが表示されます。

メッセージ画面に表示された内容を保存しておきたい場合は、「ログをクリップボードにコピー」をクリックしてから、テキストエディタなどにペースト(貼り付け)してログを保管してください。

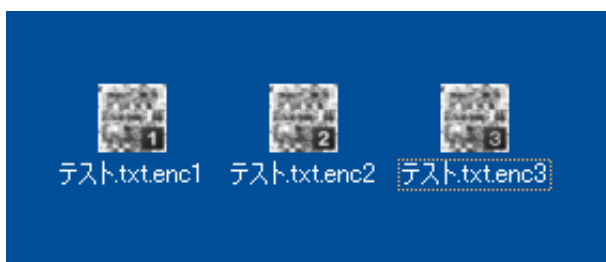
また、暗号化処理の終了後には、次のような拡張子のファイルが生成されます。

- ・ 暗号化を行った場合には、拡張子 .enc のファイルが生成されます。
- ・ 秘密分散を行った場合には、拡張子 .enc1 .enc2 .enc3 のファイルが生成されます。

◆暗号化を行った場合



◆暗号化と秘密分散を行った場合



フォルダオプションにて「登録されている拡張子は表示しない」設定にしている場合は、「.enc」「.enc1」「.enc2」「.enc3」は画面に表示されません。

復号化-1（本ソフトウェアの起動）

暗号化したファイルを元に戻すには、暗号化時と同じようにして本ソフトウェアを起動します。
(暗号化されたファイル(拡張子が enc,enc1,enc2,enc3 のファイル)を、デスクトップ上の「マル秘分散」アイコンにドラッグ&ドロップする、もしくは、右クリックの「送る」メニューから「マル秘分散」を選択します)

復号化-2（メイン画面）

下図のような画面が起動しますので、パスワードを入力後に実行ボタンをクリックします。

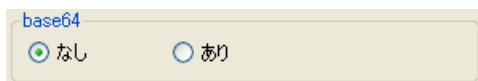
なお、動作オプションは下記のようにご指定ください。

◆暗号アルゴリズム



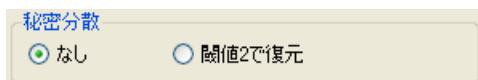
暗号化時と同じ値を指定します。

◆base64



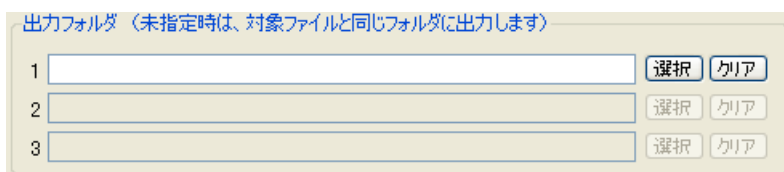
暗号化時と同じ値を指定します。

◆秘密分散



暗号化時に「なし」を選択した場合は、「なし」を選択し、
暗号化時に「なし」以外を選択した場合は、「閾値 2 で復元」を選択します。

◆出力フォルダ



暗号化時と同様に、ファイルを出力するフォルダを指定することができます。
なお、復号化時は、フォルダ-1しか指定できません。

復号化-3 (処理終了後)

復号化処理の終了後には、暗号化処理終了時と同等のメッセージ画面が表示されます。

また、暗号化・秘密分散時に付加された「.enc」「.enc1」「.enc2」「.enc3」などの拡張子がカットされたファイル名(暗号化を行う前のファイル名)で、ファイルが出力されます。

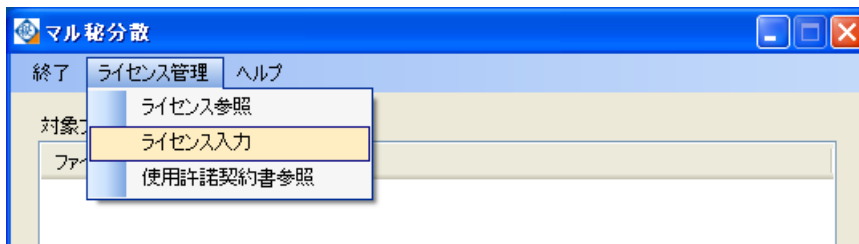
ライセンスの入力

※重要:ライセンスの入力は、管理者権限のあるユーザーで行ってください。

本ソフトウェアを、試用期間の経過後も引き続きご使用いただくためには、ライセンスをご購入いただき、下記のようにしてライセンスキーをご入力いただく必要があります。

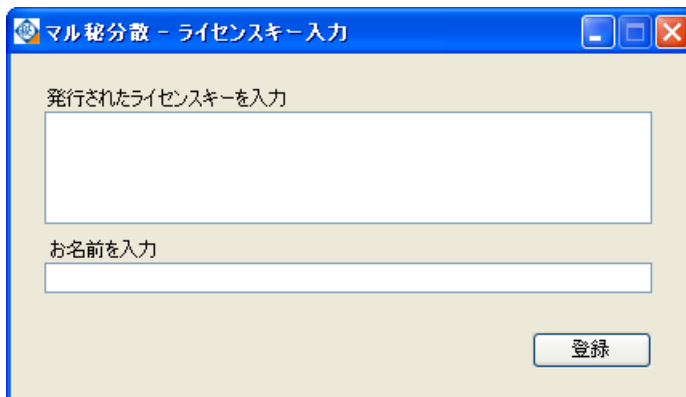
◆ライセンス入力手順-1

下図のように、上部メニューから「ライセンス管理」→「ライセンス入力」を選択します。



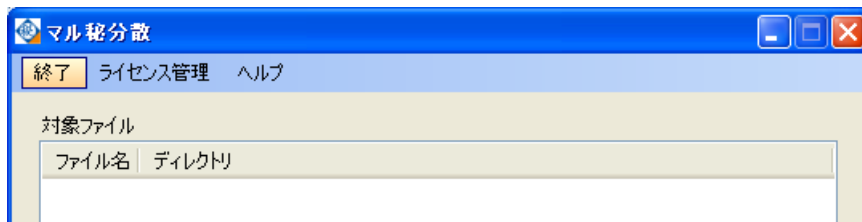
◆ライセンス入力手順-2

次のような画面が起動しますので、ご購入いただいたライセンスキーとお名前をご入力ください。

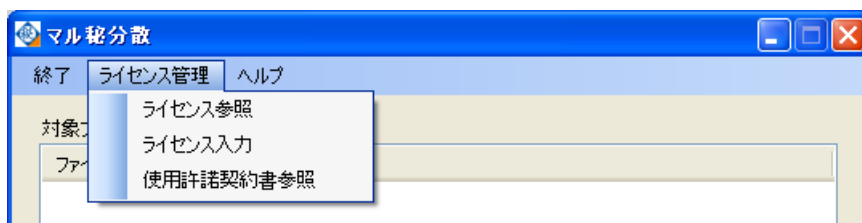


その他の機能

画面上部のメニューから、下記の操作を行うことが可能です。

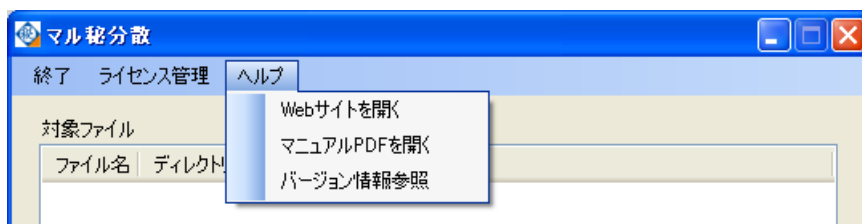


上図の終了を押下すると、プログラムが終了します。



上図の「ライセンス管理」から下記の3操作が可能です。

- ・ ライセンス参照:本ソフトウェアに入力されている、ライセンス情報を参照できます。
- ・ ライセンス入力:前ページ解説のように、ライセンスキーの入力を行えます。
- ・ 使用許諾契約書参照:使用許諾契約書の内容を参照できます。



上図の「ヘルプ」から下記の3操作が可能です。

- ・ Web サイトを開く:ブラウザから「マル秘分散 Web 版」へのアクセスが行われます。
- ・ マニュアル PDF を開く:マニュアルの PDF ファイルが開かれます。
- ・ バージョン情報参照:本ソフトウェアのバージョンが表示されます。

よくある質問

暗号化時に指定したパスワードを忘れてしまったのですが。

暗号化時に指定されたパスワードをお忘れの場合には、復号化を行うことができませんため、パスワードは確実に管理して頂けますようお願いいたします。

また、暗号アルゴリズムなどの各オプションの指定値を、暗号化時と異なる値にして復号化を行うこともできません。

暗号化/秘密分散したファイルの拡張子について

本ソフトウェアでは、元のファイル名に .enc を付加したファイル名で暗号化ファイルを作成します。復号化においては、暗号化ファイル名から .enc を削除したファイル名で復号化ファイルを作成します。

秘密分散した場合は、.enc の後に数値を付加しています。

例)

「a.txt」を暗号化すると、「a.txt.enc」が生成されます。

「a.txt.enc」を復号化すると、「a.txt」として復元されます。

「a.txt」を 2/2 秘密分散すると、「a.txt.enc1」と「a.txt.enc2」が生成されます。

「a.txt」を 2/3 秘密分散すると、「a.txt.enc1」と「a.txt.enc2」と「a.txt.enc3」が生成されます。

「a.txt」を 2/2 秘密分散した場合、「a.txt.enc1」と「a.txt.enc2」の両方が復元に必要です。

「a.txt」を 2/3 秘密分散した場合、復元のためには下記のいずれかの組み合わせでファイルが必要になります。

「a.txt.enc1」と「a.txt.enc2」

「a.txt.enc1」と「a.txt.enc3」

「a.txt.enc2」と「a.txt.enc3」

暗号化と秘密分散を同時に行うとどうなるのですか？

本ソフトウェアで暗号化&秘密分散を行うと、元ファイルを暗号化した後、その暗号化ファイルに対して秘密分散を行います。

暗号化/秘密分散をしておけば安全ですか？

しないよりは、基本的には安全です。

ただし、暗号化/秘密分散はセキュリティの一部であり全てではありません。

例えば、いくらデータを暗号化していても、コンピュータにウイルスやスパイウェアが進入している場合には、パスワードを盗み見される可能性もあります。

また、悪意のある侵入者にとって、暗号化されたファイルは「このファイルにはきっと有益な情報がふくまれているだろう。」という想像の対象になるため、パスワードの管理がおろそかな暗号化ファイルは逆に危険な状況を生み出す可能性もあります。

秘密分散したファイルの保管方法は？

分散した各ファイルを別々の箇所に保管することが望ましいです。

例えば2枚に分散した場合、1枚をPCに保管し、もう1枚をUSBメモリなどに保管すれば、どちらか1つが盗難にあっても元ファイルの復元はできません。

その他の秘密分散のメリットは？

例えば、東京支店から大阪支店にCD-Rでデータを送付する必要がある場合、分散した各ファイルを別郵便で送付すれば、どちらか1つが盗難にあっても元ファイルの復元はできません。

制約事項

非常に大きなファイルを、base64 オプション「あり」で秘密分散する場合、メモリが足りずに処理が異常終了する場合があります。base64 オプション「なし」で秘密分散する場合はこの問題が発生しません(本ソフトウェアが、お客様の性能要件を満たせるかどうかは、本ソフトウェア試用期間内にご確認を頂けますようお願い致します)。

フォルダをまるごと暗号化することには対応しておりますが、デバイスをまるごと暗号化することはできません。また、特殊なファイルやそれを含むフォルダを暗号化することはできません(お客様が作成した通常のファイルを、暗号化・秘密分散することには問題ありません)。

他のプロセス(プログラム)が開いているファイルは正しく処理できない場合があります。

ごく稀に、暗号化時と異なるパスワードを入力しても、復号化において警告が表示されない場合がございます(ただし、その場合、復号化によって生成されたファイルの内容は、暗号化前のもとは異なる意味のないバイナリデータとなっています)。

コマンドモードの使用方法（上級者・開発者向け）

コマンドモードについて、はじめに

※本ページ以降の解説は、サーバー処理やバッチ処理を想定とした、上級者・開発者向けのものになります。通常ユーザーの方には、本ページ以降の解説をお読みいただく必要はありません。

本ソフトウェアは、フォーム画面からの操作によるオペレーションの他に、コマンドモードによるオペレーションも提供しています。

コマンドモードをご使用いただくことで、暗号化・復号化・秘密分散の各処理を、コマンドプロンプトや独自のアプリケーションからご利用頂くことが可能となります。

※独自のアプリケーションからご利用頂く場合にも、ご利用者数に応じたライセンス数のご購入が必要となりますのでご注意ください。また、サーバー環境にて本ソフトウェアのコマンドモードをご利用になる場合には、クライアント版とは別途のライセンスが必要となります。

必要なライセンス数などについてご不明な点などございましたら、info@slogical.co.jp までご連絡を頂けますようお願いいたします。

コマンドモードのご使用方法

コマンドモードで使用するコマンドは、プログラムのインストールフォルダ¥cmd¥hibun.exe となります。（通常、C:¥Program Files¥Slogical Corporation¥Maruhi Bunsan¥cmd¥hibun.exe となります）

次のように環境変数 PATH をセットすると、hibun.exe がコマンドプロンプト等から実行可能となります。
set PATH=%PATH%;C:¥Program Files¥Slogical Corporation¥Maruhi Bunsan¥cmd

また、コマンドモードでは、標準入力の各行にパラメータを入力することで、動作オプションの指定を行います（詳しくは、次ページ以降の解説をご参照ください）。

コマンドモードでの処理が正常に終了した場合は、hibun.exe は終了コード 0 を返し、異常終了した場合は終了コード 1 を返しますので、独自のアプリケーション等から hibun.exe ご利用の際には終了コードの確認処理も実装して頂けますようお願いいたします。

暗号化する

hibun.exe を実行して、標準入力から次のように入力します。

1行目	暗号化コマンド名(下記参照)
2行目	暗号アルゴリズム(下記参照)
3行目	パスワード
4行目	処理対象ファイル名(暗号化したいファイルのファイル名)
5行目	出力ファイル名(ここで指定したファイル名の暗号化ファイルが生成されます)

暗号化コマンド名には、下記のいずれかの値を指定します。

通常の暗号化を行う場合	enc
暗号化を行い、base64して出力する場合	enc_b64
暗号化を行い、Salted_ を付けずに出力する場合	enc_nomagic
暗号化を行い、Salted_ を付けずにbase64して出力する場合	enc_b64_nomagic

暗号アルゴリズムには、下記のいずれかの値を指定します。

AES256bit で暗号化する場合	aes-256-cbc
AES192bit で暗号化する場合	aes-192-cbc
AES128bit で暗号化する場合	aes-128-cbc
Camellia256bit で暗号化する場合	camellia-256-cbc
Camellia192bit で暗号化する場合	camellia-192-cbc
Camellia128bit で暗号化する場合	camellia-128-cbc

復号化する

hibun.exe を実行して、標準入力から次のように入力します。

1行目	復号化コマンド名(下記参照)
2行目	暗号アルゴリズム(下記参照)
3行目	パスワード
4行目	処理対象ファイル名(暗号化したいファイルのファイル名)
5行目	出力ファイル名(ここで指定したファイル名で復号されます)

復号化コマンド名には、下記のいずれかの値を指定します。

enc で暗号化した場合	dec
enc_b64 で暗号化した場合	dec_b64
enc_nomagic で暗号化した場合	dec_nomagic
enc_b64_nomagic で暗号化した場合	dec_b64_nomagic

暗号アルゴリズムには、暗号化時に指定した時と同じものを指定します。

秘密分散(分散する)

hibun.exe を実行して、標準入力から次のように入力します。

1行目	秘密分散コマンド名(下記参照)
2行目	復元しきい値(整数で指定)
3行目	分散枚数(整数で指定)
4行目	処理対象ファイル名(分散したいファイルのファイル名)
5行目以降	出力ファイル名(分散枚数の数だけ繰り返し入力する)

秘密分散コマンド名には、下記のいずれかの値を指定します。

通常秘密分散を行う場合	ss
秘密分散を行い、base64して出力する場合	ss_b64

※復元しきい値には、分散枚数で指定した値よりも小さな値を指定する必要があります。

※例えば、分散枚数を 3 とした場合、次のように合計で7行の標準入力を行う必要があります。

- ・ 5行目:分散ファイル(1/3)のファイル名を入力
- ・ 6行目:分散ファイル(2/3)のファイル名を入力
- ・ 7行目:分散ファイル(3/3)のファイル名を入力

秘密分散(復元する)

hibun.exe を実行して、標準入力から次のように入力します。

1行目	秘密分散コマンド名(下記参照)
2行目	復元しきい値(整数で指定)
3行目	出力ファイル名
4行目以降	入力ファイル名(分散されたファイルを、復元しきい値の数だけ繰り返し入力)

秘密分散コマンド名には、下記のいずれかの値を指定します。

ss で秘密分散した場合	sr
ss_bb64 で秘密分散した場合	sr_b64

※復元しきい値には、分散時に指定した値と同じものを指定します。

※例えば、復元しきい値を 2 とした場合、次のように合計で5行の標準入力を行う必要があります。

- ・ 4行目:分散ファイル-1 のファイル名を入力
- ・ 5行目:分散ファイル-2 のファイル名を入力

Base64 エンコード&デコード

hibun.exe を実行して、標準入力から次のように入力します。

1行目	Base64 コマンド名(下記参照)
2行目	処理対象ファイル名
3行目	出力ファイル名(ここで指定したファイル名で出力されます)

復号化コマンド名には、下記のいずれかの値を指定します。

Base64 エンコードする場合	b64enc
Base64 デコードする場合	b64dec

独自アプリケーション内から、Base64 処理を単体で呼び出したい場合などにご使用ください。

ファイルのハッシュ値参照

hibun.exe を実行して、標準入力から次のように入力します。

1行目	hash (固定)
2行目	ハッシュアルゴリズム(下記参照)
3行目	処理対象ファイル名(ハッシュ値を参照したいファイルのファイル名)

ハッシュアルゴリズムには、下記のいずれかの値を指定します。

md5 でハッシュする場合	md5
sha1 でハッシュする場合	sha1
sha256 でハッシュする場合	sha256
sha384 でハッシュする場合	sha384
sha512 でハッシュする場合	sha512
tiger でハッシュする場合	tiger
ripemd160 でハッシュする場合	ripemd160
whirlpool でハッシュする場合	whirlpool

上記の入力後、ハッシュ値が標準出力されます。

ファイルの強力削除

hibun.exe を実行して、標準入力から次のように入力します。

1行目	delfile (固定)
2行目	処理対象ファイル名(削除したいファイルのファイル名)

※上記を実行すると、ファイルに対して下記の処理を行います。

- ・ ファイルに対し、ファイルサイズと同じサイズのランダムデータを書き込む
- ・ ファイルに対し、ファイルサイズと同じサイズのランダムデータをもう一度書き込む
- ・ ファイルに対し、ファイルサイズと同じサイズのゼロデータを書き込む
- ・ ファイルを削除する

本コマンドは、ファイルの削除時に、ハードディスクの未使用領域にデータを残りにくくすることを目的として実装されておりますが、ファイルシステムの特性上、ハードディスクの未使用領域から完全にデータが削除されることを保証するものではありません。

独自のアプリケーションから hibun.exe をご利用になる際などで、一時的な仮ファイルを比較的安全に削除する必要がある際には、本コマンドをご利用ください。

※「ごみ箱」を空にするなど通常の方法でファイルの削除を行うと、OS からは瞬時にファイルが見えなくなりますが、ハードディスクの未使用領域にはデータが残ったままになっています(その後、OS を使い続けると、その未使用領域にデータが上書きされ、次第にハードディスクからデータが完全に消えていきます)。

ライセンスキーの入力

hibun.exe を実行して、標準入力から次のように入力します。

1行目	license_write (固定)
2行目	お客様のお名前
3行目以降	ライセンスキー

※重要: 本コマンドは、管理者権限のあるユーザーで行ってください。

ライセンス情報の参照

hibun.exe を実行して、標準入力から次のように入力します。

1行目	license_read (固定)
-----	-------------------

ライセンス情報が標準出力されます。

Linux 版をお使いのお客様(フォーム画面からライセンス情報を参照できないお客様)は、このコマンドによってライセンス情報を確認できます。

なお、製品版をお使いの場合は、標準出力されたライセンス情報の1文字目が「b」で始まります。「b」で始まらない場合は、ライセンスキーの入力に失敗している可能性があります。

サンプルアプリケーション

Perl(Active Perl)から hibun.exe を呼び出すサンプルプログラムは下記のとおりです。

```
---- ここから -----
#!/usr/bin/perl

## -----
## マル秘分散コマンドモード用サンプルスクリプト ver1.0.0 (Perl)
## Copyright (C) 2008 Slogical Corporation. All Rights Reserved.
##
## 下記の環境変数をセットしてから実行してください
## <Windows 版>
## set PATH=%PATH%;C:\Program Files\Slogical Corporation\Maruhi Bunsan\cmd
## <Linux 版>
## export PATH=$PATH:/(インストールディレクトリ)/bin/
##
## 作業フォルダ (フォルダ名は「test」) を作成して、その中で各コマンドを実行します。
## -----

use strict;
use IPC::Open2;

## hibun.exe 実行用のサブルーチン
## -----
sub do_hibun {
    my @cmd = @_ ;
    my $buff = "";

    my $pid = open2(*R, *W, "hibun.exe");
    for my $cmd (@cmd) {
        print W $cmd . "\n";
    }
    close(W);
    while (<R>) {
        $buff .= $_;
    }
    close(R);

    waitpid($pid, 0);
    if ($? != 0) {
        print "FAILED.\n";
    }

    return $buff;
}

## 作業フォルダの作成
## -----
my $workdir = "test";
if (-d $workdir || -f $workdir) {
    print STDERR "$workdir" is already found.\n";
    exit(0);
}
system("mkdir $workdir");
chdir($workdir) or die("can not change dir to $workdir.");

## テスト用ファイル作成
```

```

## -----
print "\n-----> make test file\n";
my @delfile;
my $testfile = "test.txt";
open(FILE, "> $testfile");
print FILE "THIS IS TEST STRING.";
close(FILE);
push(@delfile, $testfile);

## ハッシュ値参照
## -----
my %hashval;
print "\n-----> hash\n";
for my $algo ("md5", "sha1", "sha256", "sha384", "sha512", "tiger", "ripemd160", "whirlpool") {
    print "$algo";
    for (my $i=0; $i < (12 - length($algo)); $i++) {
        print " ";
    }

    my $sum = do_hibun("hash", $algo, $testfile);
    $sum =~ s|[\r\n]+$||;
    $hashval{$algo} = $sum;

    print $sum . "\n";
}

## 暗号化・復号化
## -----
print "\n-----> enc/dec\n";
my $cnt_enc = 0;
my $cnt_dec = 0;
for my $algo ("aes-256-cbc", "aes-192-cbc", "aes-128-cbc", "camellia-256-cbc", "camellia-192-cbc",
"camellia-128-cbc") {
    my $pass = "testpass";

    my $encfile          = $testfile . ". " . $algo . ". normal" . ". enc";
    my $encfile_b64      = $testfile . ". " . $algo . ". b64" . ". enc";
    my $encfile_nomagic  = $testfile . ". " . $algo . ". nomagic" . ". enc";
    my $encfile_b64_nomagic = $testfile . ". " . $algo . ". b64_nomagic" . ". enc";
    my $decfile          = $testfile . ". " . $algo . ". normal" . ". txt";
    my $decfile_b64      = $testfile . ". " . $algo . ". b64" . ". txt";
    my $decfile_nomagic  = $testfile . ". " . $algo . ". nomagic" . ". txt";
    my $decfile_b64_nomagic = $testfile . ". " . $algo . ". b64_nomagic" . ". txt";
    push(@delfile, $encfile);
    push(@delfile, $encfile_b64);
    push(@delfile, $encfile_nomagic);
    push(@delfile, $encfile_b64_nomagic);
    push(@delfile, $decfile);
    push(@delfile, $decfile_b64);
    push(@delfile, $decfile_nomagic);
    push(@delfile, $decfile_b64_nomagic);

    do_hibun("enc",          $algo, $pass, $testfile,          $encfile);
    do_hibun("dec",          $algo, $pass, $encfile,          $decfile);
    do_hibun("enc_nomagic", $algo, $pass, $testfile,          $encfile_nomagic);
    do_hibun("dec_nomagic", $algo, $pass, $encfile_nomagic, $decfile_nomagic);
    do_hibun("enc_b64",     $algo, $pass, $testfile,          $encfile_b64);
    do_hibun("dec_b64",     $algo, $pass, $encfile_b64,      $decfile_b64);
    do_hibun("enc_b64_nomagic", $algo, $pass, $testfile,          $encfile_b64_nomagic);
    do_hibun("dec_b64_nomagic", $algo, $pass, $encfile_b64_nomagic, $decfile_b64_nomagic);

    $cnt_enc += 4;
}

```

```

        $cnt_dec += 4;
    }
    print "enc: $cnt_enc files¥n";
    print "dec: $cnt_dec files¥n";

## 秘密分散
## -----
print "¥n-----> ss/sr¥n";
my $cnt_ss = 0;
my $cnt_sr = 0;
my $tm_ss = 0;
my $tm_sr = 0;
for (my $i=1; $i<=10; $i++) {
    my $cmd_ss = ($i % 2 == 1) ? "ss" : "ss_b64";
    my $cmd_sr = ($i % 2 == 1) ? "sr" : "sr_b64";

    my $ssfile_21 = $testfile . ".ss" . $i . ".2.enc1";
    my $ssfile_22 = $testfile . ".ss" . $i . ".2.enc2";
    my $ssfile_31 = $testfile . ".ss" . $i . ".3.enc1";
    my $ssfile_32 = $testfile . ".ss" . $i . ".3.enc2";
    my $ssfile_33 = $testfile . ".ss" . $i . ".3.enc3";

    my $srfile_2 = $testfile . ".sr" . $i . ".2.txt";
    my $srfile_31 = $testfile . ".sr" . $i . ".31.txt";
    my $srfile_32 = $testfile . ".sr" . $i . ".32.txt";
    my $srfile_33 = $testfile . ".sr" . $i . ".33.txt";

    push(@delfile, $ssfile_21);
    push(@delfile, $ssfile_22);
    push(@delfile, $ssfile_31);
    push(@delfile, $ssfile_32);
    push(@delfile, $ssfile_33);
    push(@delfile, $srfile_2);
    push(@delfile, $srfile_31);
    push(@delfile, $srfile_32);
    push(@delfile, $srfile_33);

    do_hibun($cmd_ss, 2, 2, $testfile, $ssfile_21, $ssfile_22);
    do_hibun($cmd_ss, 2, 3, $testfile, $ssfile_31, $ssfile_32, $ssfile_33);

    do_hibun($cmd_sr, 2, $srfile_2, $ssfile_21, $ssfile_22);
    do_hibun($cmd_sr, 2, $srfile_31, $ssfile_31, $ssfile_32);
    do_hibun($cmd_sr, 2, $srfile_32, $ssfile_31, $ssfile_33);
    do_hibun($cmd_sr, 2, $srfile_33, $ssfile_32, $ssfile_33);

    $cnt_ss += 5;
    $cnt_sr += 4;
    $tm_ss += 2;
    $tm_sr += 4;
}
print "ss: $cnt_ss files ($tm_ss times)¥n";
print "sr: $cnt_sr files ($tm_sr times)¥n";

## Base64
## -----
print "¥n-----> Base64¥n";
my $cnt_b64_enc = 0;
my $cnt_b64_dec = 0;
for (my $i=1; $i<=10; $i++) {
    my $encfile = $testfile . ".b64" . $i . ".b64";
    my $decfile = $testfile . ".b64" . $i . ".txt";

```



```

push(@delfile, $encfile);
push(@delfile, $decfile);

do_hibun("b64enc", $testfile, $encfile);
do_hibun("b64dec", $encfile, $decfile);

$sct_b64_enc++;
$sct_b64_dec++;
}
print "b64_enc: $sct_b64_enc files¥n";
print "b64_dec: $sct_b64_dec files¥n";

## 本ディレクトリ内の全 txt ファイルの sha256 値を参照
## -----
print "¥n-----> check sum (sha256) for [dec|sr|b64_dec] files¥n";
my $sct_ok = 0;
my $sct_ng = 0;
opendir(DIR, ".");
my @file = readdir(DIR);
closedir(DIR);
for my $file (@file) {
    next if ($file !~ m|¥.txt$| || $file eq $testfile);
    my $sum = do_hibun("hash", "sha256", $file);
    $sum =~ s|[¥r¥n]+$||;

    if ($sum eq $hashval{"sha256"}) {
        $sct_ok++;
    } else {
        $sct_ng++;
    }
}
print "OK: $sct_ok files¥n";
print "NG: $sct_ng files¥n";

## 強力削除
## -----
my $do_del = 0;
$| = 1;
print "¥n¥nDO YOU WANT TO REMOVE WORK FILES ? [y/n] ";
while (<STDIN>) {
    $_ =~ s|[¥r¥n]+$||;
    if ($_ eq "y" || $_ eq "Y" || $_ eq "n" || $_ eq "N") {
        $do_del = 1 if ($_ eq "y" || $_ eq "Y");
        last;
    } else {
        print "DO YOU WANT TO REMOVE WORK FILES ? [y/n] ";
    }
}
if ($do_del == 1) {
    my $sct_del = 0;
    print "¥n-----> delfile¥n";
    for my $delfile (@delfile) {
        do_hibun("delfile", $delfile);
        $sct_del++;
    }
    print "del: $sct_del files¥n";

    chdir("../");
    rmdir($workdir);
}

## 終了

```

```
## -----  
exit(0);
```

```
--- ここまで -----
```