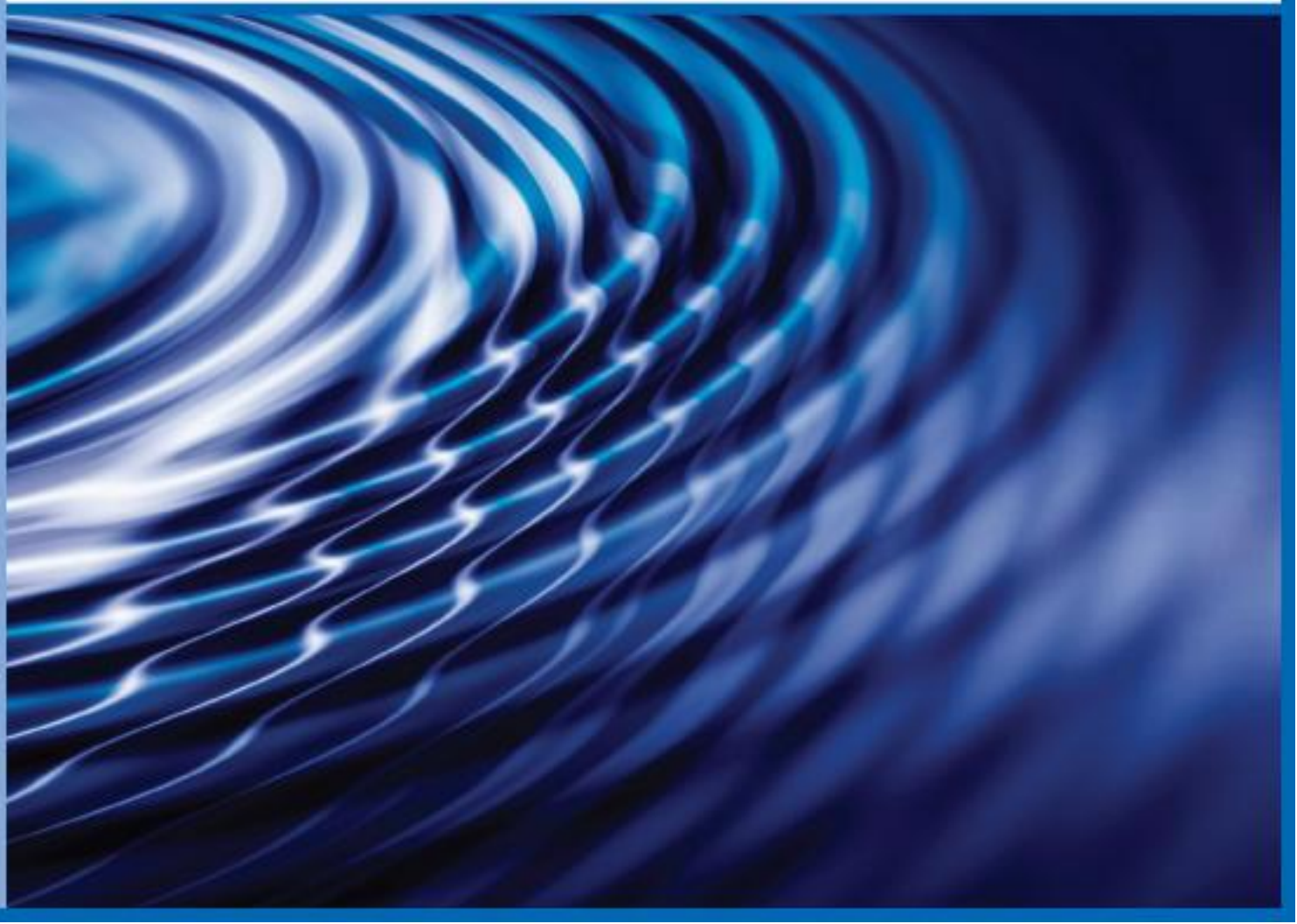
WINDOWS クライアント ユーザガイド

Attachmate® Reflection®

for Secure IT



RSIT 7.2 SP3

ユーザガイド

Reflection for Secure IT

Windows クライアント

バージョン 7.2 SP3

2013 年 04 月 30 日 印刷

Attachmate Corporation
705 5th Avenue South
Seattle, WA 98104
USA
+1.206.217.7100
http://www.attachmate.com

目次

概要

Reflection for Secure IT Windows クライアント は、完全な機能を備えたカスタマイズしやすい Windows ベースの Secure Shell クライアントです。セキュリティが確保されていないネットワーク上で、信頼するホストと Windows ワークステーション間で安全な暗号化通信を提供します。使用しているローカルコンピュータとリモートホスト（複数可）間のすべての接続が暗号化され、これらのマシン間で送信されるデータは保護されます。Telnet、FTP、rlogin、または rsh を使用する場合、パスワードがプレーンテキスト形式のままネットワークを介して送信されることは絶対にありません。

Reflection for Secure IT Windows クライアント は、以下に対応しています。

- プロトコルバージョン 1 サーバおよびプロトコルバージョン 2 サーバへの安全な接続。
- 次のような標準の Secure Shell 機能:TCP ポート転送 (X-11 を含む)、データストリームの圧縮と暗号化、認証 (パスワード、キーボード対話型、公開鍵、または Kerberos/GSSAPI)、およびログ記録。
- RSA、RSA1、および DSA 鍵を作成できるユーザ鍵生成ツール。
- Secure Shell サーバに公開鍵をアップロードするためのツール。Reflection は、自動的にサーバの種類を検出し、正しい鍵の種類をエクスポートし、サーバ上の正しい位置にインストールします。
- 信頼されるホスト鍵を表示し、管理するためのツール。
- 単一のパスフレーズで複数の鍵と証明書を管理し、追加のサーバに認証を転送できる鍵エージェントユーティリティ。
- Reflection 固有の証明書格納場所で証明書を管理できる証明書マネージャを含む PKI サポート。また、Windows 格納場所、スマートカード、または他の PKCS#11 準拠ハードウェアデバイスにある証明書を使用するように、Reflection を構成できます。
- 安全な SFTP ファイル転送。
- ssh、ssh-keygen、sftp、および scp 用のスタンドアロンの DOS コマンドラインユーティリティ。

1 章

インストール

この章の内容

対応するプラットフォームの詳細と追加のシステム要件については、*技術ノート 1944* 『*http://support.attachmate.com/techdocs/1944.html*』を参照してください。

Reflection for Secure IT は、通常、電子的に配布されます。インストール CD が必要な場合は、注文時に CD を要求する必要があります。

ワークステーションへの Reflection for Secure IT のインストール

注意: Reflection for Secure IT をインストールするには管理者権限でログオンする必要があります。必要なアクセス権限がない場合は、システム管理者に権限の変更を依頼してください。

ワークステーションにインストールするには

1 Attachmate インストールプログラムを実行します。

インストール元	手順
ダウンロードサイト	ダウンロードリンクをクリックして、ダウンロードしたプログラムを実行します。インストーラファイルの場所を選択して **[次へ]** をクリックします。これにより、ファイルが指定の場所に解凍され、Attachmate インストールプログラムが起動します。
管理者用インストールイメージ	管理者用インストールポイントから、setup.exe ファイルをダブルクリックします。

2 Attachmate インストールプログラムから **[続行]** をクリックし、ライセンスに同意します。

3 (オプション) 特定個人用にインストールを設定するには、**[ユーザ情報]** タブをクリックし、名前、組織、ボリューム購入契約 (VPA) 番号 (VPA がある場合) を入力します。

注意: VPA 番号は、サービス要求を迅速に処理できるようにカスタマーサポートが使用します。

4 (オプション) 既定のインストールフォルダを変更するには、**[ファイルの場所]** タブをクリックし、Reflection for Secure IT をインストールするフォルダを参照します。

5 (オプション) インストールされる機能、コンポーネント、言語を選択するには、**[機能の選択]** タブをクリックします。

6 **[インストール]** をクリックします。

注意: **[詳細設定]** タブは、管理者用インストールを作成する場合にのみ使用します。 管理インストールでは、製品はワークステーションにインストールされません。管理者用インストールポイントにファイルがコピーされます。このネットワーク上の場所は、配布ツールがワークステーションに配布するパッケージにアクセスして作成するのに使用します。エンドユーザはこの場所にある setup.exe を実行して、ワークステーションへのインストールを実行することができます。

機能および言語の選択

[機能の選択] タブを使用して、製品機能のインストール方法を選択します。

インストールする機能、コンポーネント、および言語を選択するには

1 **[機能の選択]** タブをクリックします。

2 アイテムごとに、以下のオプションから選択します。

選択	目的
[機能をローカルのハードディスクドライブにインストールする]	アイテムをインストールします。
[機能を必要時にインストールする]	アイテムをアドバタイズします。例えば、[スタート] メニューからコンポーネントを選択できます。その際にインストールされます。
[機能を使用不可にする]	アイテムをアンインストールされたままにしておきます。Windows の [プログラムと機能] (または [プログラムの追加と削除]) コントロールパネルを使用して、後でアイテムをインストールできます。

以前のバージョンからのアップグレード

バージョン 7.2 にアップグレードする前に、次の情報を確認してください。

- 以前のバージョンをアンインストールする必要はありません。このバージョンをインストールすると、インストーラが自動的に古いバージョンを検出し、アップグレードします。既存の Secure Shell 設定は有効のまま残り、既存の設定ファイルをそのまま使用できます。
- このバージョンをインストールすると、以前のバージョンの Reflection for Secure IT を保持できません。別のインストール場所を選択する場合でも、インストーラは自動的に古いバージョンをアップグレードします。アップグレードによって以前のバージョンは削除されます。
- このバージョンをインストールした後で削除しても、以前のバージョンには戻りません。

2 章

はじめに

この章の内容

Secure Shell は、リモートコンピュータへのログインとコマンドの実行を安全に行うためのプロトコルです。これは、Telnet、FTP、rlogin、あるいは rsh の代わりとなる安全な方法です。Secure Shell 接続では、ホスト (サーバ) とユーザ (クライアント) の両方の認証が必要です。また、ホスト間の通信はすべて暗号化された通信チャネルを介して行う必要があります。また、Secure Shell では X11 セッションまたは指定の TCP/IP ポートを、安全なトンネルを介して転送することもできます。

新しい端末セッションの開始

ほとんどの場合、既定の設定を変更せずに、ホストに接続し、パスワードを使用してログオンすることができます。

既定値を使用して新しい端末セッションを開始するには

1 Windows の [スタート] メニューで、**[Attachmate Reflection]** - **[SSH クライアント]** をクリックします。

2 Reflection for Secure IT ツールバーで、**[接続/切断]** ボタンをクリックします。

3 **[ホストに接続]** ダイアログボックスにホストとユーザ名を入力し、**[OK]** をクリックします。

注意:初めてこのホストに接続している場合は、ホストの認証情報の確認を求めるダイアログボックスが表示されます。ホストのシステム管理者に問い合わせることで、ホスト鍵の有効性を確認できます。このホストを既知のホスト一覧に追加するには **[常時]** をクリックします。

---

4 このホスト用のパスワードを入力し、**[OK]** をクリックします。

5 このセッション構成とともに設定ファイルを保存するには、**[ファイル]** - **[保存]** をクリックします。

構成ツールバーの表示

構成ツールバーを使用すると、迅速にセッション設定にアクセスできます。

構成ツールバーを表示するには

1 Reflection for Secure IT セッションを開く

2 **[セッション設定の構成]** ツールバーボタンをクリックします。

3 セッションを開くたびに、セッション設定を保存する（**[ファイル]** - **[保存]**）ための、このツールバーが表示されます。

FTP クライアントを使用したファイルの転送

ドラッグしてドロップするだけで、ファイルを FTP クライアントに転送できます。個々のファイル、複数のファイル、およびフォルダ全体をドラッグできます。

サーバに接続し、ファイルを転送するには

1 Windows の [スタート] メニューで、**[Attachmate Reflection]** - **[FTP クライアント]** をクリックします。**[FTP サイトに接続]** ダイアログボックスが自動的に開きます。

2 **[新規]** をクリックします。

3 ウィザードの指示に従って進み、プロンプトが表示されたら、ホストとユーザ名を入力します。

注意：FTP クライアントと Reflection for Secure IT をインストールする場合、ウィザードは既定で SFTP 接続を作成するように構成されています。**[セキュリティのプロパティ]** ダイアログボックスを使用すると、追加の接続の種類を構成することができます。**[ログイン情報]** ダイアログボックスにある **[セキュリティ]** ボタンを使用します。

---

4 最後のパネルで、ホストに接続するかどうか確認を求められます。既定では、**[はい]** が選択されます。この選択のまま、**[完了]** をクリックしてウィザードを終了し、接続を確立します。

注意：まだ接続していないサーバとの SFTP 接続を確立している場合は、ホストの認証情報の確認を求めるダイアログボックスが表示される場合があります。ホストのシステム管理者に問い合わせることで、ホスト鍵の有効性を確認できます。このホストを既知のホスト一覧に追加するには **[常時]** をクリックします。

---

5 転送するファイルまたはフォルダおよび転送先を検索して見つけます。

検索先	実行する場所
ローカルフォルダ	左画面
サーバディレクトリ	右画面

6 転送するファイルまたはフォルダを選択して、転送元の場所から目的の転送先へドラッグします。

このサーバをサイトの一覧に保存するには

- **[ファイル]** - **[保存]** をクリックして、FTP クライアント設定ファイルにサイト構成を保存します。

保存したサイトに接続するには

1 FTP クライアントを起動します。

2 **[FTP サイトに接続]** ダイアログボックスで、サイトをクリックして選択します。

3 **[接続]** をクリックします。

注意：FTP クライアントの操作に関する完全な情報については、FTP クライアントアプリケーションヘルプを参照してください。

SSH クライアントセッションから Secure Shell 設定を構成するには

1 **[接続]** メニューで **[接続の設定]** をクリックします。

2 **[接続オプション]** から、**[ホスト名]** および（オプションで）**[SSH 構成セクション]** の値を入力します（**[SSH 構成セクション]** を空白のまま残すと、行った変更は、**ホスト名**と同じ名前を持つ *SSH 構成セクション*『116ページ 』に保存されます）。

3 **[セキュリティ]** をクリックします。

注意：ホスト名を入力しないと **[セキュリティ]** ボタンは使用できません。

FTP クライアント から Secure Shell 設定を構成するには

1 **[FTP サイトに接続]** ダイアログボックスでサイトをクリックして選択します。

2 **[セキュリティ]** をクリックします。

3 **[Secure Shell]** タブから、**[Reflection Secure Shell を使用する]** を選択します（FTP クライアント を Reflection for Secure IT とともにインストールする場合、これは既定で選択されます）。

4 （オプションで）**[SSH 構成セクション]** を指定します（**[SSH 構成セクション]** を空白のまま残すと、Secure Shell 設定は、ホスト名と同じ名前を持つ *SSH 構成セクション*『116ページ 』に保存されます）。

5 **[構成]** をクリックします。

[全般] タブ（[Reflection Secure Shell の設定] ダイアログボックス）

表示方法『15ページ 』

次のオプションがあります。

[ポート番号]	サーバ上の接続先ポートを指定します。既定値（22）は、Secure Shell 接続の標準ポートです。
[プロトコル]	ホストへの接続の確立時に Reflection が使用する Secure Shell プロトコルのバージョンを指定します。この設定で最も安全な値は **[2 のみ]** です。

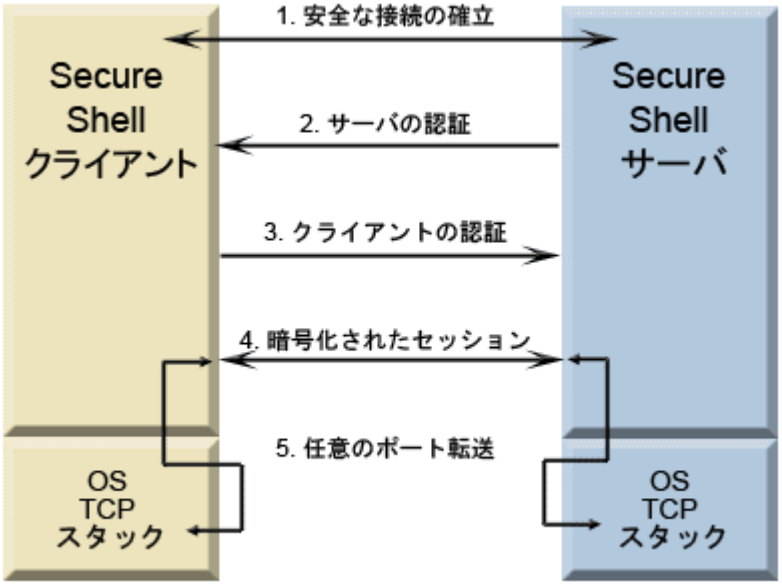
[ユーザ認証]	いずれかの*認証方式*『27ページ 』の横にあるボックスをクリックして、その方式をオフにするか、オンにします。少なくとも 1 つの認証方式を選択する必要があります。プロトコル 2 接続の場合は、矢印キーを使って優先順位を指定できます。Reflection は、それぞれの方式を先頭から順番に試行します。
[サーバキープアライブ]	**[サーバキープアライブ]** チェックボックスをオンにすると、Reflection は、指定された間隔で、安全なトンネルを介して NOOP メッセージをサーバへ送信します。この設定は、サーバとの接続を維持するために使用します。**[間隔]** を使用すると、サーバアライブメッセージを送信する頻度を指定できます。この設定が有効になっていないと、サーバが停止するかネットワーク接続が失われた場合に Secure Shell 接続が終了されません。この設定は、TCP セッションのみを転送する接続がサーバによって時間切れになるのを防ぐためにも使用できます。サーバは、SSH トラフィックが存在しないことの検出を理由にこれらの接続を時間切れにする場合があるからです。 Secure Shell の **[サーバキープアライブ]** 設定は、すべての TCP/IP 接続がファイアウォールによって時間切れになるのを防ぐために Windows レジストリに設定可能な TCP キープアライブ設定と関連はありません。TCP/IP キープアライブの動作を変更するには、Windows レジストリを編集する必要があります。
[圧縮を使用]	**[圧縮を使用]** チェックボックスをオンにすると、クライアントはすべてのデータの圧縮を要求します。圧縮はモデム回線やほかの低速接続に適していますが、高速のネットワークでは応答速度の低下を招くだけです。圧縮レベルの設定はプロトコルバージョン 1 でのみ可能で、プロトコルバージョン 2 接続では適用されません。
[既存の接続がある場合、それを利用する]	既定で、同一ホストへの複数のセッションでは元の Secure Shell 接続が*再利用*『28ページ 』されます。このため、再認証が不要となります。**[既存の接続がある場合、それを利用する]** チェックボックスをオフにすると、Reflection はセッションごとに新しい接続を確立します。つまり、新しい接続ごとに認証プロセスが繰り返されます。
[記録内容]	Secure Shell の*ログファイル*『76ページ 』に書き込む情報量を決定します。

注意

- このダイアログボックスで構成した設定は、*Secure Shell 構成ファイル*『19ページ 』に保存されます。また、任意のテキストエディタを使用して、このファイルを編集して Secure Shell 設定を構成することもできます。
- この構成ファイルの内の設定は、現在指定されている *SSH 構成セクション*『116ページ 』用に保存されます。

Secure Shell とは

ここでは、データを安全に送信するための Secure Shell チャネルの作成と使用に関する基本手順の概要を記載します。



1. **安全な接続を確立します。**

   クライアントとサーバは、セッションの暗号化に使用する共有鍵と暗号、およびデータの完全性保証の確認に使用するハッシュを作成するために交渉します。

2. **サーバを認証します。**

   サーバ認証によって、クライアントはサーバの ID を確認できます。 サーバからクライアントへの認証は、この認証プロセス中に 1 回だけ可能です。この認証に失敗した場合は、接続できません。

3. **クライアントを認証します。**

   クライアント認証によって、サーバはクライアントユーザの ID を確認できます。 既定で、クライアントは認証を複数回試行できます。サーバとクライアントは、1 つまたは複数の認証方式に合意するように交渉します。

4. **暗号化されたセッションを介してデータを送信します。**

   暗号化されたセッションが確立されると、Secure Shell サーバとクライアント間で交換されるすべてのデータが暗号化されます。この段階で、ユーザはサーバへの安全なリモートアクセスが可能になり、保護されたチャネルを通じて安全にコマンドを実行し、ファイルを転送することができます。

5. **ポート転送を使用して、その他のクライアントとサーバ間の通信を保護します。**

   トンネリングとしても知られるポート転送は、アクティブなセッションにおける Secure Shell チャネルを通じて通信をリダイレクトするための方法を提供します。ポート転送が構成されると、指定のポートへ送信されるすべてのデータは、保護されたチャネルを通じてリダイレクトされます。

3 章

構成

この章の内容

Reflection for Secure IT および FTP クライアント は、設定を保存するために多くのファイル使用しています。

設定ファイル

設定ファイルは、アプリケーション固有の設定を構成します。Reflection for Secure IT と FTP クライアントは、異なる設定ファイルを使用します。

アプリケーション	拡張子	構成対象
Reflection for Secure IT	*.r3w	ホスト接続情報、端末エミュレーション、表示設定、キーの割り当て、マウスの構成
FTP クライアント	*.rfw	ホスト接続情報、ディレクトリ表示設定、転送設定

設定ファイルを保存するには、**[ファイル]** - **[保存]** を使用します。

注意:これらの設定の構成については、Reflection for Secure IT および FTP クライアントのアプリケーションヘルプを使用してください。この情報は、このユーザガイドでは扱っていません。

Secure Shell クライアント構成ファイル

Secure Shell 構成ファイルには、Secure Shell クライアント接続に固有の設定が含まれています。このユーザ固有ファイルは、*[Reflection Secure Shell の設定] ダイアログボックス*『15ページ 』を使用して設定を変更する時に自動的に作成され、更新されます。設定は、このダイアログボックスを閉じると自動的に保存されます。ファイル名および場所は次のとおりです。

<個人用ドキュメントフォルダ>¥Attachmate¥Reflection¥.ssh¥config

このファイルの設定はホストごと（または *SSH 構成セクション*『116ページ 』ごと）に適用され、Reflection for Secure IT クライアントおよび FTP クライアントの両方に影響します。例えば、Reflection for Secure IT を使用して、Acme.com に対する接続に既定以外の Secure Shell 設定を構成すると（SSH 構成セクションは指定しません）、Secure Shell 設定は次の行に示されるセクションの構成ファイルに保存されます。

Host Acme.com

Acme.com に接続するように FTP クライアントも構成した場合 (SSH 構成セクションは指定しません)、FTP クライアントは構成ファイルの「Host Acme.com」セクションの設定を使用します (設定は、両方のアプリケーションで同じ SSH 構成セクションを指定した場合と同じ方法で共有されます)。

注意: **[Reflection Secure Shell の設定]** ダイアログボックスを閉じる時、既定の設定値は構成ファイルに保存されません。既定値を手動でファイルに追加している場合、ダイアログボックスを閉じる時にその既定値は削除されます。ワイルドカードホストのスタンザを、特定のホスト名を使用するスタンザと組み合わせて使用する場合、これにより、設計制約が課せられます。ワイルドカードのスタンザで構成された値を上書きするよう設定された特定のホストのスタンザに既定値を手動で構成した場合、ホスト固有の SSH 構成セクションの設定を表示するために [Reflection Secure Shell 設定] ダイアログボックスを開くと、既定値は削除されます。この状況は、グローバル構成ファイルを使用すると適切に処理することができます。グローバル構成ファイルは、ユーザが [Reflection Secure Shell の設定] ダイアログボックスを開いたり閉じたりしても、更新されません。

グローバル構成ファイル

システム管理者は、システム規模の構成ファイルをインストールすることもできます。ファイル名および場所は次のとおりです。

<アプリケーションデータフォルダ>『145ページ 』¥Attachmate¥Reflection¥ssh_config

このファイルの設定は、コンピュータのすべてのユーザのクライアント接続に影響します。

4 章

暗号化

この章の内容

サポートされている暗号化アルゴリズム

データ暗号化の規格

暗号化は、転送中のデータの機密性を保護します。 この保護はデータ送信前に、秘密鍵と暗号を使用してデータを暗号化することで実現します。 受信データは、同じ鍵と暗号を使用して解読する必要があります。 特定のセッション用に使用される暗号は、クライアントの優先順位が最高の暗号であり、サーバもこの一覧に対応しています。

Reflection for Secure IT Windows クライアントは、以下のデータ暗号化規格に対応しています。

- DES (56 ビット) - SSH プロトコル 1 でのみ使用可能
- Arcfour、Arcfour128、および Arcfour258 (ストリームモード)
- TripleDES (168 ビット) CBC モード
- Cast (128 ビット)
- Blowfish (128 ビット) CBC モード
- AES (別名 Rijndael) (128、192、または 256 ビット) CBC モードおよび CTR モード

[データ整合性]

データ整合性は、データが転送中に変更されていないことを保証します。Secure Shell 接続は、データの整合性を保証するために MAC (メッセージ認証コード) を使用します。クライアントとサーバはそれぞれ個別に、転送されたデータの各パケットのハッシュを計算します。メッセージが転送中に変更された場合、ハッシュの値は異なり、パケットが却下されます。特定のセッション用に使用される MAC は、クライアントの優先順位が最高の MAC であり、サーバもこの一覧に対応しています。Reflection は、次の MAC 規格をサポートしています。

- hmac-sha1
- hmac-md5
- hmac-sha1-96
- hmac-md5-96
- hmac-ripemd-160
- hmac-sha256
- hmac-sha2-256
- hmac-sha512
- hmac-sha2-512

[デジタル署名]

デジタル署名は、公開鍵認証（証明書認証を含む）に使用されます。認証機関はデジタル署名を使用して、認証を受ける団体が正しい秘密鍵を保持していることを確認します。Secure Shell クライアントはデジタル署名を使用して、ホストを認証します。公開キー認証が構成されると、Secure Shell サーバはデジタル署名を使用してクライアントを認証します。Reflection は、次のデジタル署名アルゴリズムをサポートしています。

- x509v3-rsa2048-sha256
- x509v3-sign-rsa
- x509v3-sign-dss
- ssh-rsa-sha2-256@attachmate.com
- ssh-rsa
- ssh-dss

連邦情報処理規格（FIPS）

FIPS モードで動作するように Reflection を構成すると、アメリカ合衆国政府の連邦情報処理規格（FIPS）140-2 が強制されます。使用できるすべての設定が、この規格に適合するセキュリティプロトコルとアルゴリズムを使用します。これらの規格に適合しないオプションは使用できません。個々のセッションが FIPS モードで動作するように構成することも、すべての Reflection セッションに FIPS モードを強制することもできます。

特定の Secure Shell セッションを FIPS モードで動作するように構成する

次の手順を使用して、特定の Secure Shell セッションを FIPS モードで動作するように構成することができます。

注意：この手順は、すべての Secure Shell セッションに FIPS 規格を強制するわけではありません。この変更は *Secure Shell 構成ファイル*『19ページ 』に保存され、特定の *SSH 構成セクション*『116ページ 』に適用されます（セクションを指定しない場合、設定は現在のホストへのすべての接続に適用されます）。以後の Secure Shell セッションが同じ SSH 構成セクション（またはホスト名）を使用するように構成されていない場合、この変更は以後の Secure Shell セッションには影響しません。

特定のホストまたは SSH 構成セクションに FIPS モードを設定するには

1 *[Secure Shell の設定] ダイアログボックスを開きます*『15ページ 』。

2 **[暗号化]** タブで、**[FIPS モードで実行する]** を選択します。

また、Secure Shell 構成ファイルを手動で編集することで、この設定を手動で設定することもできます。FIPS モードを設定するためのキーワードは **FIPSMode** です。

すべての Reflection セッションを FIPS モードで動作するように構成する

管理者は、Reflection グループポリシーを使用して、全ての Reflection セッションが FIPS モードで動作するように構成できます。

すべてのセッションに FIPS モードを設定するには

1 グループポリシーエディタを実行するには、次のいずれかの方法を使用します。

- コマンドラインで次のように入力します。

  Gpedit.msc

- **[Active Directory ユーザとコンピュータ]** コンソールで **[組織単位]** のプロパティを開き、**[グループポリシー]** タブをクリックして新規ポリシーオブジェクトを編集または作成します。

2 *Reflection テンプレート (ReflectionPolicy.adm) がインストールされていない場合はインストールします。*『93ページ 』

3 **[ローカルコンピュータポリシー]** - **[ユーザの構成]** - **[管理用テンプレート]** - **[Reflection Settings]** で、**[Allow non-FIPS mode] (FIPS 以外のモードを許可する)** 設定を無効にします。

[暗号化] タブ ([Reflection Secure Shell の設定] ダイアログボックス)

表示方法『15ページ 』

[Reflection Secure Shell の設定] ダイアログボックスの **[暗号化]** タブを用いて、Secure Shell 接続で使用される*暗号*『146ページ 』を指定します。接続に使用する Secure Shell プロトコルに応じて、異なるオプションが使用可能になります。

次のオプションがあります。

[SSH プロトコル 2]

[暗号一覧]

この一覧は、現在のホストとのプロトコル 2 接続に使用する*暗号*『146ページ 』を指定するために使用します。複数の暗号を選択すると、Secure Shell クライアントは、指定した順番で先頭から暗号の使用を試みます。順序を変更するには、一覧から暗号を選択して上向きまたは下向きの矢印をクリックします。特定のセッション用に使用された暗号は、この一覧の最初のアイテムであり、サーバもこの一覧に対応しています。

[HMAC 一覧]

使用可能にする HMAC (ハッシュメッセージ認証コード) 方式を指定します。このハッシュは、サーバと交換されるすべてのデータパケットの整合性を検証するために使用されます。複数の HMAC を選択すると、Secure Shell クライアントは、指定した順番で先頭の HMAC からサーバと交渉します。順序を変更するには、一覧から HMAC を選択して上向きまたは下向きの矢印をクリックします。

[鍵交換アルゴリズム]

クライアントが対応する鍵交換アルゴリズムと、その優先順位を指定します。

場合によっては、DH Group14 SHA1 を他の値より前に配置するために、鍵交換アルゴリズムの順番を変更する必要があります。hmac-sha512 MAC を使用する場合、または鍵交換中に次のエラーが表示される場合は、この操作が必要です。"fatal:

dh_gen_key: group too small:1024 (2*need 1024)"

その他に 2 つの暗号化アルゴリズム (gss-group1-sha1-*) に対応していますが、これらは使用可能な鍵交換アルゴリズムの一覧には表示されません。これらの 2 つのアルゴリズムは、**[全般]**『15ページ 』タブ (**[ユーザ認証]** の下) で GSSAPI/Kerberos を使用可能にし、**[GSSAPI]**『56ページ 』タブで **[Reflection Kerberos]** をオンにすると、クライアントによって自動的に提案されます。

[指紋の種類] 秘密鍵の所有を証明する過程でクライアントが使用するハッシュアルゴリズムを指定します。公開鍵ユーザ認証時にこのハッシュが使用されます。RSA 鍵で使用されるハッシュを指定するには **RSA** を使用し、DSA 鍵で使用されるハッシュを指定するには **DSA** を使用します。

[SSH プロトコル 1 暗号] この設定は、現在のホストとのプロトコル 1 接続に使用したい暗号を選択するために使用します。既定値は [Triple DES] で、これが推奨されるオプションです。

[FIPS モードで実行する] [FIPS モードで実行する] をオンにすると、Reflection は、その接続に対してアメリカ合衆国政府の連邦情報処理標準 (FIPS) 140-2 を施行します。[FIPS モードで実行する] をオンにした場合、この標準に適合していない [暗号化] タブのオプションは使用不可になります。

注意

- このダイアログボックスで構成した設定は、*Secure Shell 構成ファイル*『19ページ 』に保存されます。また、任意のテキストエディタを使用して、このファイルを編集して Secure Shell 設定を構成することもできます。
- この構成ファイルの内の設定は、現在指定されている *SSH 構成セクション*『116ページ 』用に保存されます。

---

5 章

認証

この章の内容

認証は、通信相手の身元を確実に確認する処理のことです。身元の確認は、パスワードなどの既知の情報、秘密鍵やトークンなど所有しているもの、または指紋などの固有の情報を使用して行います。 Secure Shell 接続では、ホスト（サーバ）とユーザ（クライアント）の両方の認証が必要です。既定で、ホストは秘密鍵を使用してユーザを認証し、ユーザはパスワードを使用してホストを認証します。

公開鍵を使用したサーバ認証

Reflection for Secure IT では、公開鍵と証明書（公開鍵認証の特別な形）という 2 種類のサーバ認証に対応しています。

ホスト認証のために公開鍵認証が使用される時は、以下の一連のイベントが行われます。

1. Secure Shell クライアントが接続を開始します。
2. サーバが公開鍵をクライアントに送信します。
3. クライアントが、信頼されているホスト鍵ストアからこの鍵を検索します。

クライアントによる鍵の検索結果	発生するイベント
ホスト鍵を見つけ、クライアントコピーが、サーバによって送信された鍵に一致する	認証は次の段階に進みます。
ホスト鍵を検出 *できない*	クライアントは、ホストが不明であるというメッセージを表示し、ホスト鍵の指紋を提供します。ユーザが不明な鍵を受け入れることができるようにクライアントが構成されている場合（既定）、ユーザは、この鍵を受け入れることができ、認証は次の段階に進みます。 厳格なホスト鍵の確認が履行される場合は、クライアントによって接続が終了されます。
ホスト鍵を見つけたが、クライアントコピーが、サーバによって送信された鍵に一致*しない*	クライアントは、鍵が既存の鍵に一致しないという警告を表示し、サーバによって送信された鍵の指紋を表示します。ユーザが不明な鍵を受け入れることができるようにクライアントが構成されている場合（既定値）、ユーザは、この新しい鍵を受け入れることができます。 厳格なホスト鍵の確認が履行される場合は、クライアントによって接続が終了されます。

4. サーバが、受信した公開鍵に対応する秘密鍵を実際に保持していることを確認するために、クライアントはサーバに試行（任意のメッセージ）を送信して、当該メッセージテキストに基づきハッシュ『146ページ 』を計算します。
5. サーバは、試行メッセージに基づきデジタル署名を作成します。これを行うために、サーバは別個にメッセージのハッシュを計算し、次に、サーバの秘密鍵を使用して、この計算したハッシュを暗号化します。サーバは、当該デジタル署名を元の試行に付加して、この署名付きのメッセージをクライアントに返します。
6. クライアントは、公開鍵を使用して署名を復号化し、クライアント自身が計算したハッシュと当該ハッシュを比較します。値が一致すると、ホスト認証が成功します。

証明書を使用したサーバ認証

証明書認証は、公開鍵認証が持つ問題の一部を解決します。公開鍵ホスト認証の場合、システム管理者は、各クライアントの既知のホスト一覧にすべてのサーバのホスト公開鍵を追加する必要があります。または、不明なホストに接続する時にホストの識別情報を確認することをクライアントユーザに依存することになります。証明書認証は、認証局（CA）と呼ばれる信頼されたサードパーティを使用して、ホストから来る情報の正しさを確認することでこの問題を防止します。

公開鍵認証と同様、証明書認証は、公開鍵/秘密鍵ペアを使用してホストの識別情報を確認します。ただし、証明書認証の場合、公開鍵は*デジタル証明書*『147ページ 』に含まれ、この場合、2 つの鍵ペアが使用されます。ホストが 1 つの秘密鍵を保持し、CA が 2 番目の公開鍵を保持します。ホストは、CA から証明書を取得します。この証明書には、ホストに関する識別情報、ホスト公開鍵のコピー、および CA の秘密鍵を使用して作成された*デジタル署名*『146ページ 』が含まれています。この証明書は、認証プロセス中にクライアントに送信されます。ホストから送信された情報の完全性を確認するため、クライアントは CA ルート証明書に封印されている CA の公開鍵のコピーを持つ必要があります。

ホストの識別情報を確認するために CA ルート証明書をインストールすることは、ホスト公開鍵をインストールし、構成することに比べて、いくつかの利点があります。

- 単一の CA 証明書を使用して、複数のサーバを認証することができます。
- 管理者は、Windows グループポリシーを使用して、Windows クライアントに CA 証明書をインストールすることができます。
- 商業的に取得した証明書のルート証明書は、既にクライアント コンピュータで使用できる場合があります。Windows コンピュータでは、Internet Explorer で使用するための、一部のルート証明書があらかじめインストールされています。SSL/TLS 接続の場合、Reflection は、この証明書格納場所に既定で置かれた証明書をチェックします。
- 必要なら、ホストは、クライアントシステムを何も変更することなく、同じ CA から新しい証明書を取得できます。

サーバ証明書認証は、次のような順序で実行されます。

1. Secure Shell クライアントが接続を開始します。
2. ホストは、その証明書をクライアントに送信します。
3. クライアントは、CA ルート証明書を使用してサーバ証明書の有効性を確認します。

注意：クライアントは、信頼されるルート格納場所に、すでに CA 証明書のコピーを持っている必要があります（単一の CA 証明書を使用して、複数のサーバを認証することができます）。

---

4. クライアントは、ホストの証明書に記載されているサーバ情報が対象ホストと一致することを確認します。
5. ホストが証明書の公開鍵に対応する秘密鍵を保持していることを確認するため、クライアントは試行（任意のメッセージ）をサーバに送信し、このメッセージ文に基づいてハッシュ『146ページ 』を計算します。

6. サーバは、試行メッセージに基づいてデジタル署名を作成します。サーバは独立してメッセージハッシュを計算し、その秘密鍵を使用して計算されたハッシュを暗号化します。次に、サーバは、このデジタル署名を試行に添付し、この署名付きメッセージをクライアントに返信します。
7. クライアントは、サーバの公開鍵を使用して署名を解読し、元のハッシュと自分で計算したハッシュを比較します。値が一致する場合、ホスト認証は成功します。

注意:Reflection クライアントは、*Reflection 証明書格納場所*『50ページ 』または Windows 証明書格納場所を使用して、ホスト証明書を確認できます。

クライアント認証方式

Reflection Secure Shell クライアントでは、4 つの方式のユーザ認証に対応しています。それらは、Kerberos (GSSAPI)、公開鍵、キーボード対話型、パスワードです。**[Reflection Secure Shell の設定]**『15ページ 』ダイアログボックスを使用して、認証の設定を構成します。少なくとも 1 つの認証方式を選択する必要があります。複数の方式を選択した場合、Secure Shell クライアントは指定した順番で認証を試行します。既定で、Reflection は最初に公開鍵認証を試行し、次にキーボード対話型、パスワードの順に試行します。

注意: 公開鍵および GSSAPI/Kerberos V5 認証方式では、サーバとクライアントの両方の構成が必要になります。

認証方式	説明
[パスワード]	クライアントユーザに、Secure Shell サーバホスト上の当該ユーザ用のログインパスワードを入力するよう求めます。 パスワードは、暗号化されたチャネルを介してホストに送信されます。
[キーボード対話型]	単純なパスワード認証を含む、キーボードを使用して認証データを入力する手順に対応します。これによって、Secure Shell クライアントは、RSA SecurID トークンまたは RADIUS サーバなどのさまざまな認証機構に対応できるようになります。 例えば、クライアント管理者はキーボード対話型認証を構成して、パスワード更新などの複数のプロンプトが必要な状況を処理できます。 キーボードデータは、暗号化されたチャネルを介してホストに送信されます。
[公開鍵]	*公開/秘密鍵ペア*『146ページ 』を信頼します。公開鍵認証を構成するには、各クライアントユーザが、鍵ペアを作成して、公開鍵をサーバにアップロードする必要があります。鍵がパスフレーズで保護されている場合は、公開鍵認証を使用して接続を完了する目的で、クライアントユーザは当該パスフレーズを入力するように求められます。
[GSSAPI/Kerberos]	Kerberos は、クライアント認証とサーバ認証の両方の代替メカニズムを提供するセキュリティプロトコルです。Kerberos 認証は、KDC (Key Distribution Center) と呼ばれる信頼されたサードパーティに依存しています。Secure Shell プロトコルは、GSSAPI (Generic Security Services Application Programming Interface) を介して Kerberos 認証に対応しています。

Secure Shell セッションにおける接続の再利用

接続を再利用することによって、すでに確立されている Secure Shell 接続に別の Secure Shell セッションを追加できます。 この簡単な例が光ファイバケーブルで、外側のパイプで接続を行い、さまざまな光ファイバストランド（セッションとトンネル）がルーティングされます。追加セッションには、新しい Reflection Secure Shell 端末セッション、新しい Reflection SFTP ファイル転送セッション、転送された X11 接続、SSH トンネルを介したポート転送用に構成された通信、または Reflection Secure Shell コマンドラインユーティリティの 1 つを使用して確立された接続などがあります。

確立されている Secure Shell 接続を再利用する場合、認証処理を繰り返す必要はありません。 新しいセッションでは、最初の接続に構成されたすべての Secure Shell 設定を必ず使用します。 認証方式、暗号または MAC 設定の違い、あるいはポート転送定義は無視されます。

Reflection ユーザインタフェースを使用して行われるすべての Secure Shell 接続では、接続の再利用が既定で有効になっています。 [Reflection Secure Shell の設定] ダイアログボックスの **[全般]**『15ページ 』タブにある **[既存の接続がある場合、それを利用する]** チェックボックスをオフにして、この機能を無効にできます。

[既存の接続がある場合、それを利用する] をオンにして接続を確立すると、以下のすべての条件に合致する場合、以降の Secure Shell セッションでは確立されている接続を再利用します。

- 新しいセッションのホスト名は、確立されている接続のホスト名と完全に一致している必要があります。
- 新しいセッションのユーザ名は、確立されている接続のユーザ名と完全に一致している必要があります。
- 新しいセッションのポート番号は、確立されている接続のポート番号と同じである必要があります（既定ではこの条件が真になります）。
- 元のセッションがホスト名とは異なる *SSH 構成セクション*『116ページ 』を使用するよう構成されている場合、新しいセッションでは同じセクションを使用するよう構成する必要があります。

注意：SSH 接続にコマンドラインユーティリティを使用している場合、既存の接続を再利用するための追加条件を満たしている必要があります。 追加条件の概要を以下に示します。

Reflection コマンドラインセッションにおける接続の再利用

接続の再利用は、Secure Shell 接続が要求され、クライアントと単一のサーバ間に多数の単純な操作が必要になり、認証と鍵交換の間隔が全接続時間のかなりの部分を占めるようなコマンドライン操作に適しています。 これは、複数の小さなファイルの転送、または大量の出力を返さない簡単なオペレーティングシステムコマンドの実行が必要な場合などです。 このような場合には、**ssh**（または **ssh2**）コマンドラインユーティリティを使用して元の SSH 接続を作成してから、以降のコマンドラインユーティリティ操作で接続を再利用するのが便利な場合があります。

既定では、Reflection Secure Shell client コマンドラインユーティリティ（**ssh**『96ページ 』、**scp**『108ページ 』、**sftp**『104ページ 』、**ssh2**『100ページ 』、**scp2**『111ページ 』、**sftp2**『108ページ 』）の接続の再利用は無効になっています。これらのコマンドユーティリティのいずれかで接続の再利用を有効にするには、以下のいずれかの方法を使用する必要があります。

- 各コマンドラインにスイッチ「-o ConnectionReuse=yes」を追加します。 最初の接続を確立し、以降のすべてのコマンドラインユーティリティで最初の接続を再利用する場合、このスイッチを使用する必要があります。 例えば、以下のコマンドを使用すると、**sftp** 接続が **ssh** コマンドで確立した接続を再利用します。

```
ssh "-o connectionReuse=yes" myuser@myhost
sftp "-o connectionReuse=yes" myuser@myhost
```

- DOS コマンドウィンドウ（またはバッチスクリプトファイルの最初）で、環境変数 SSHConnectionReUse を次のとおり設定します。

  ```
  set SSHConnectionReUse=yes
  ```

矛盾する設定が存在する場合、-o スイッチが優先されます。

注意：

- OpenSSH サーバは、未認証の同時セッション数を制限するために使用できる MaxStartups パラメータに対応しています。 この設定は、既存の接続を再利用する、確立可能な Reflection セッション数に影響します。 MaxStartups パラメータに指定した最大セッション数に到達すると、以降のすべてのセッションで個別の SSH 接続と認証が必要になります。現在許可されている数より多くの未認証の同時セッションを確立する必要がある場合は、ssh サーバ管理者に問い合わせてください。
- コマンドラインユーティリティでは、Secure Shell *構成ファイル* 『19ページ 』に接続再利用を構成できません。 このファイルのキーワード **ConnectionReuse** は、スイッチ **-H** を使用してこの設定を含む *SSH 構成セクション* 『116ページ 』を指定しても、Reflection コマンドラインユーティリティによって必ず無視されます。

---

6 章

公開鍵認証

この章の内容

公開鍵認証は、公開/秘密鍵のペアに依存します。公開鍵認証は、サーバ（ホスト）とクライアント（ユーザ）の両方の認証に使用できます。Secure Shell クライアントに公開鍵認証を構成するには、クライアントに鍵のペアを作成（またはインポート）し、公開鍵をホストにアップロードします。[Reflection Secure Shell の設定] ダイアログボックスの **[ユーザ鍵]**『35 ページ 』タブまたは Reflection 鍵エージェントのいずれかを使用して、クライアント認証用の公開鍵を作成および管理できます。鍵の構成方法に応じて、公開鍵認証を使用して接続を完了するために*パスフレーズ*『146ページ 』の入力を求められる場合があります。

公開鍵認証の 1 つの形式は、X.509 証明書を使用して行われます。*Reflection 証明書マネージャ*『49ページ 』あるいは Windows 証明書マネージャによって管理される証明書を使用して認証するように Reflection を構成できます。公開鍵認証は、認証に証明書を使用する場合に有効にする必要があります。

公開鍵認証の仕組み

公開鍵暗号では、公開/秘密鍵ペアと数値アルゴリズムを併用して、データの暗号化および復号化を行います。鍵の片方は公開鍵で、これは通信相手に自由に配布できます。もう片方の鍵は秘密鍵で、鍵の所有者が安全に保管しておく必要があります。秘密鍵によって暗号化されたデータは公開鍵によってのみ復号化でき、公開鍵によって暗号化されたデータは秘密鍵によってのみ復号化できます。

鍵が認証に使用される時は、認証される側のユーザが、公開/秘密鍵ペアの秘密鍵を使用してデジタル署名を作成します。受信者は、対応する公開鍵を使用して、このデジタル署名の信頼性を確認する必要があります。これは、受信者が、他方のユーザの公開鍵のコピーを所有し、その鍵の信頼性を信頼しなければならないことを意味します。

ユーザ鍵の管理

このセクション内

公開鍵認証の構成

以下の手順では、公開鍵を使用してクライアント認証を構成します。

クライアントに公開鍵認証を構成するには手順に従います。

1 **[Reflection Secure Shell の設定]**『15ページ 』ダイアログボックスを開きます。

2 **[全般]** タブで、**[ユーザ認証]** の **[公開鍵]** がオンになっていることを確認します（公開鍵認証のみを使用する場合は、その他のオプションをオフにします）。

3 **[ユーザ鍵]** タブをクリックします。**[使用]** 列で、現在指定されているホストへの認証に使用する鍵を 1 つまたは複数選択します。

注意： 鍵をこの一覧に追加するには、「*ユーザ鍵一覧への鍵の追加*『32ページ 』」を参照してください。

---

4 **[OK]** をクリックします。

サーバに公開鍵認証を構成するには手順に従います。

- *公開鍵をホストにアップロードします*『33ページ 』。

ユーザ鍵一覧への鍵の追加

[Reflection Secure Shell の設定] ダイアログボックスの **[ユーザ鍵]**『35ページ 』タブは、*公開鍵*『31ページ 』認証に使用可能な鍵の一覧を表示します。 新しい鍵の生成または既存の鍵のインポートによって、一覧に鍵を追加できます。

Reflection を使用して新しい鍵のペアを生成するには

1 **[Reflection Secure Shell の設定]**『15ページ 』ダイアログボックスを開きます。

2 **[ユーザ鍵]** タブをクリックします。

3 **[鍵の生成]** をクリックします。

4 鍵の種類と長さを指定します。

5 *パスフレーズ*『146ページ 』を指定するか、**[パスフレーズなし]** をオンにします。

警告：**[パスフレーズなし]** を選択した場合、コンピュータに保存された秘密鍵は暗号化されず、この鍵にアクセスできる人は鍵を使用して認証できるようになります。

6 **[生成]** をクリックします。

既定で、鍵はユーザの .ssh フォルダに生成されます。 既定の秘密鍵は、鍵の種類、サイズ、およびクライアントホスト名を識別します。 公開鍵は、*.pub ファイル拡張子が追加された秘密鍵名を使用して、同じ場所に保存されます。

鍵エージェントを使用して新しい鍵のペアを生成するには

1 Reflection 鍵エージェントを起動してロック解除します（Windows の [スタート] メニューから、**[すべてのプログラム]** > **[Attachmate Reflection]** > **[ユーティリティ]** > **[鍵エージェント]** に進みます）。

2 **[鍵の生成]** をクリックします。

3 鍵の名前、種類、長さを指定し、**[OK]** をクリックします。

注意： 鍵エージェントを使用して作成する鍵は、暗号化された形でエージェントにより保存されます。

Reflection の鍵格納場所に鍵をインポートするには

1 **[Reflection Secure Shell の設定]**『15ページ 』ダイアログボックスを開きます。

2 **[ユーザ鍵]** タブをクリックします。

3 **[インポート]** をクリックします。

4 インポートしたい秘密鍵を検索して指定します。 鍵のペアごとに、*.pub ファイルと拡張子のないファイルの 2 つが保存されています。 秘密鍵は、拡張子のないファイルです。

注意： インポートした鍵は、ユーザの .ssh フォルダにある Reflection の鍵格納場所にコピーされます。

サーバへのクライアント公開鍵のアップロード

[ユーザ鍵] タブの **[アップロード]** ボタンを使用して、公開鍵を Secure Shell サーバにアップロードします。公開鍵は、安全な SFTP プロトコルを使用して転送されます。公開鍵をアップロードするには、パスワード認証（または別の認証方式）を使用できる必要があります。公開鍵のアップロードに成功すると、その他の認証方式を無効にできます。

鍵をアップロードするには

1 **[Reflection Secure Shell の設定]**『15ページ 』ダイアログボックスを開きます。

2 **[ユーザ鍵]** タブから鍵を選択し、**[アップロード]** をクリックします（鍵を選択していない場合または証明書を選択していない場合、**[アップロード]** ボタンを使用できません）。

3 入力を求められたら、ホスト名、認証するユーザ名、ユーザパスワードを入力します。

4 ホストへの安全な接続が確立されると、ダイアログボックスが開き、この鍵をアップロードするホスト上の場所に関する情報が表示されます。通常は、この設定を変更する必要はありません。詳細については、以下の「注意」を参照してください。

[公開鍵のアップロード] ダイアログボックスに、転送に関する情報が表示されます。

5 **[OK]** をクリックしてこのダイアログボックスを閉じます。

注意

- Reflection for Secure IT、F-Secure、および SSH Communications (SSH Tectia) サーバが実行されているホストにアップロードした鍵は、RFC 4716 準拠形式でエクスポートされます。既定で、これらの鍵はユーザの .ssh2 ディレクトリにインストールされ、適切な **Key** エントリが authorization ファイルに追加されます。このファイルがまだない場合は、新規作成され、適切なファイル権限が付与されます。
- OpenSSH サーバが実行されているホストにアップロードした鍵は、OPENSSH 形式でエクスポートされます。既定で、これらの鍵はユーザの .ssh ディレクトリにある authorized_keys ファイルに追加されます。このファイルがまだない場合は、新規作成され、適切なファイル権限が付与されます。

ユーザ鍵パスフレーズの変更

ユーザ鍵の保護に使用する*パスフレーズ*『146ページ 』を変更できます。

パスフレーズを変更するには

1 **[Reflection Secure Shell の設定]**『15ページ 』ダイアログボックスを開きます。

2 **[ユーザ鍵]** タブをクリックし、一覧から鍵を選択します。

3 **[パスフレーズの変更]** をクリックします (鍵を選択していない場合、あるいは、Reflection 証明書マネージャまたは Windows 証明書マネージャのいずれかによって管理される証明書を選択している場合、このボタンは使用できません)。

ユーザ鍵のエクスポート

次の手順で、ユーザ鍵を新しい場所または形式にエクスポートできます。

注意:Secure Shell サーバに公開鍵をアップロードする場合は、この手順を使用する必要はありません。**[アップロード]** ボタンを使用してワンステップで実行できます。Reflection は、指定したサーバの正しい鍵形式を自動的に判断します。詳細については、「*サーバへのクライアント公開鍵のアップロード*『33ページ 』」を参照してください。

鍵をエクスポートするには

1 **[Reflection Secure Shell の設定]**『15ページ 』ダイアログボックスを開きます。

2 **[ユーザ鍵]** タブで、鍵を選択し **[エクスポート]** をクリックします (鍵が選択されていない場合、または Reflection 証明書マネージャか Windows 証明書マネージャのどちらかによって管理されている証明書を選択している場合、このボタンは使用できません)。

3 選択した鍵の*パスフレーズ*『146ページ 』を入力します。

4 (オプション)

目的	手順
エクスポートに秘密鍵を含める	**[秘密鍵をエクスポートする]** をオンにします。
鍵を OpenSSH 形式でエクスポートする	**[OpenSSH 形式で保存する]** をオンにします。

5 **[公開鍵ファイル名]** ダイアログボックスでエクスポートする鍵の名前と場所を指定します。

6 **[保存]** をクリックします。

[ユーザ鍵] タブ ([Reflection Secure Shell の設定])

表示方法『15ページ 』

[ユーザ鍵] タブには、*公開鍵認証*『31ページ 』を使用して Secure Shell 接続を構築する際に、ホストに対するクライアントセッションを認証する鍵を作成および管理するためのツールがあります。

注意：**[OK]** をクリックすると、このダイアログボックスでの変更が、現在指定されている *SSH 構成セクション*『116ページ 』に保存されます。

Reflection は、使用可能なユーザ鍵を保持しています。現在のホストに対して、Reflection が認証用に使用する鍵を 1 つまたは複数指定するには、**[使用]** 列の 1 つまたは複数のチェックボックスをオンにします（または **[すべての鍵をホストの認証に使用する]** を有効にします）。

鍵の一覧には次の内容が含まれます。

- **[ユーザ鍵の生成]**『37ページ 』ダイアログボックスを使用して作成した鍵
- **[インポート]** ボタンを使用して追加した鍵
- Reflection Secure Shell フォルダに手動でコピーした鍵
- Reflection 鍵エージェント内の鍵および証明書
- F-Secure 設定から Reflection へ移行する間にコピーされたユーザおよび認証エージェントの鍵
- 個人用格納場所の Windows 証明書マネージャ内の証明書
- 個人用格納場所の *Reflection 証明書マネージャ*『49ページ 』内の証明書

次の鍵管理ツールも使用できます。

ホストの認証に使用する鍵を選択します。

[表示]	選択した鍵または証明書の内容を表示します。
[生成]	**[ユーザ鍵の生成]**『37ページ 』ダイアログボックスを開きます。このダイアログボックスは、ユーザ鍵認証用の公開鍵と秘密鍵のペアを構成するために使用できます。
[アップロード]	現在指定されているホストへ *公開鍵をアップロード*『33ページ 』します。
[インポート]	使用可能な鍵の一覧に秘密鍵を追加します。この機能を使用すると、他のアプリケーションを使用して作成した鍵に Reflection 内で簡単にアクセスできます。鍵をインポートすると、Reflection Secure Shell フォルダにコピーされます。
[エクスポート]	*公開鍵をエクスポート*『34ページ 』するか、公開鍵と秘密鍵のペアをエクスポートします。
[削除]	選択した鍵を削除します。
[パスフレーズの変更]	選択した鍵の保護に使用する *パスフレーズ*『146ページ 』を変更します。
[鍵エージェントに追加]	選択した鍵を Reflection 鍵エージェントに追加します。鍵エージェントを起動したことがない場合や、鍵エージェントがロックされている場合は、鍵エージェントのパスフレーズの入力を要求されます。また、鍵をエージェントに追加できるようにする場合は、秘

密鍵のパスフレーズの入力を要求されます。

認証オプション

[すべての鍵をホストの認証に使用する]

このオプションをオンにすると、クライアントは（[使用] チェックボックスがオンになっているかどうかに関わらず）表示されているすべての鍵を使用して認証を行います。

[証明書署名よりも SSH 鍵署名を優先する]

この設定によって、公開鍵の認証時にクライアントがサーバに提示する証明書署名の種類の順序が決定されます。この設定がオン（既定）の場合、クライアントは最初に標準の ssh 鍵署名 (ssh-rsa または ssh-dss) を使用して鍵を送信します。それに失敗すると、クライアントは再び証明書署名 (x509-sign-rsa または x509-sign-dss) の使用を試みます。

このオプションがオフの場合、クライアントは証明書署名を最初に提示します。これは、証明書鍵の種類が必要であり、サーバがクライアントに対して署名の種類が異なる同じ鍵を 2 回目の認証で使用することを許可していない場合に便利です。

鍵エージェント

[エージェントの転送を許可する]

Reflection 鍵エージェント接続の転送を有効にします。エージェントの転送を有効にする場合は注意が必要です。エージェントの UNIX ドメインソケットのリモートホストでファイル権限を回避できるユーザは、転送された接続を介してローカルエージェントにアクセスできます。攻撃者は鍵の情報を取得できませんが、エージェントに読み込まれた識別情報を使用して、その鍵で操作を実行して認証を有効にすることができます。

[ホストの認証に使用する鍵を鍵エージェントに追加する]

この設定は、**[エージェントの転送を許可する]** が有効な場合に使用できます。これがオンになっており、サーバに対する公開鍵認証が成功すると、認証に使用された鍵または証明書が自動的に Reflection 鍵エージェントに追加されます。この鍵は鍵エージェントには保存されませんが、鍵エージェントが稼働中は使用可能な状態のままです。

[鍵エージェント起動]

Reflection 鍵エージェントを起動します。

[ユーザ鍵の生成] ダイアログボックス

表示方法

1 **[Reflection Secure Shell の設定]** ダイアログボックスを開きます。

2 **[ユーザ鍵]** タブをクリックします。

3 **[鍵の生成]** をクリックします。

このダイアログボックスは、ユーザ鍵認証用に公開/秘密鍵のペアを構成するために使用します。

次のオプションがあります。

[鍵の種類] 鍵の生成に使用する鍵のアルゴリズムを指定します。

[鍵の長さ] 鍵のサイズを指定します。鍵のサイズを大きくすると、ある程度までセキュリティは向上します。鍵のサイズを大きくすると最初の接続が遅くなりますが、正常に接続した後は、鍵のサイズはデータストリームの暗号化や解読の速度に影響しません。使用する鍵の長さは、多くの要素に依存します。その要素には、鍵の種類、鍵の有効期間、保護するデータの値、潜在的な攻撃者にとって利用可能なリソース、この非対称鍵とともに使用する対称鍵のサイズなどがあります。ニーズに合った最適な選択をするには、セキュリティ管理者にお問い合わせください。

[パスフレーズなし] パスフレーズの入力を求められることなく接続したい場合は、このチェックボックスをオンにします。 注意： **[パスフレーズなし]** チェックボックスをオンにした場合、コンピュータに保存されている秘密鍵は、暗号化されません。

[パスフレーズ] この鍵を使った接続時に要求される*パスフレーズ*『146ページ 』を指定します。 注意：パスフレーズを使用しない場合は、**[パスフレーズなし]** チェックボックスをオンにする必要があります。

[確認] 確認のためにパスフレーズを再入力します。

[生成] 参照ダイアログボックスが開き、秘密鍵の名前と場所を選択できます(既定の名前は、鍵の種類、サイズ、およびクライアントホスト名を識別します)。 公開鍵は、*.pub ファイル拡張子が追加された秘密鍵名を使用して、同じ場所に保存されます。

ホスト鍵の管理

このセクション内

ホスト鍵の確認の構成

この手順を使用して、不明なホストへの接続時に Reflection がどのように動作するかを指定します。

ホスト鍵の確認を構成するには

1 **[Reflection Secure Shell の設定]**『15ページ 』ダイアログボックスを開きます。

2 **[ホスト鍵]** タブをクリックします。

3 **[厳格なホスト鍵の確認の履行]** をクリックします。

4 次のオプションのどちらかを選択します。

選択する項目	結果
[ユーザに尋ねる] (既定値)	不明なホストに接続すると、*[ホスト鍵の信頼性]*『41ページ 』確認のダイアログボックスが表示されます。
[はい]	厳格なホスト鍵の確認を履行します。ホストが信頼するホストでない場合、Reflection は接続しません。接続前に、ホスト鍵を信頼するホスト鍵の一覧に追加する必要があります。
[いいえ]	厳格なホスト鍵の確認を履行しません。確認のダイアログボックスが表示されずに、Reflection は接続します。信頼する鍵の一覧にホスト鍵を追加しません。

注意

- ホストが X.509 証明書を使用して認証するように構成されている場合、**[厳格なホスト鍵の確認の履行]** には影響しません。ホストがホスト認証のために証明書を提示し、信頼されたルート格納場所に必要な CA 証明書がない場合、接続に失敗します。
- この設定に加えた変更は、現在指定されている *SSH 構成セクション*『116ページ 』に保存されます。
- Secure Shell 設定は、*Secure Shell 構成ファイル*『19ページ 』に保存されます。また、このファイルを任意のテキストエディタで手作業で編集することにより Secure Shell 設定を構成することもできます。この設定の構成に使用される *キーワード*『118ページ 』は **StrictHostKeyChecking** です。

優先するホスト鍵の種類の構成

[証明書よりも SSH 鍵を優先する] を使用して、ホスト鍵アルゴリズムの優先順位を指定します。この設定は、証明書と標準ホスト鍵認証の両方をサーバに構成する場合に便利です。SSH プロトコルでは、ホストの認証試行を 1 回しか許可しません。ホストが証明書を提示し、証明書を使用したホスト認証がクライアントに構成されていない場合、接続に失敗します (これは、複数の認証試行に対応しているユーザ認証とは異なります)。

優先するホスト鍵の種類 (標準 SSH 鍵または証明書) を構成するには

1 **[Reflection Secure Shell の設定]**『15ページ 』ダイアログボックスを開きます。

2 **[ホスト鍵]** タブをクリックします。

3 ホストが認証に標準ホスト鍵を使用するようにするには、**[証明書よりも SSH 鍵を優先する]** を選択します。

または

認証に証明書を使用するには、**[証明書よりも SSH 鍵を優先する]** をオフにします。

注意

- この設定に加えた変更は、現在指定されている *SSH 構成セクション*『116ページ 』に保存されます。
- Secure Shell 設定は、*Secure Shell 構成ファイル*『19ページ 』に保存されます。また、このファイルを任意のテキストエディタで手作業で編集することにより Secure Shell 設定を構成することもできます。この設定の構成に使用されるキーワードは **HostKeyAlgorithms** です。

既知のホストファイル

Reflection Secure Shell クライアントでは、既知のホストの一覧を既知のホストファイルに保存します。Reflection は、ユーザ固有のホストファイルとグローバルな既知のホストファイルの両方に対応しています。

ユーザ既知のホストファイル

ユーザ固有の既知のホストファイルは known_hosts と呼ばれ、ユーザの .ssh フォルダにあります。これは、既定の既知のホストファイルです。Reflection は、以下の場合にこのファイルを自動的に更新します。

- [Reflection Secure Shell の設定] ダイアログボックスの **[ホスト鍵]**『40ページ 』タブにある **[信頼されているホスト鍵]** 一覧を更新した場合

または

- これまで知らなかったホストに接続し、**ホスト鍵の信頼性**『41ページ 』メッセージに対して **[常時]** と応答した場合

グローバルな既知のホストファイル

システム管理者は、ssh_known_hosts という名前のシステム規模で既知のホストファイルを *Reflection アプリケーションデータフォルダ*『144ページ 』に追加できます。

この場所で、既知のホストファイルにより PC のすべてのユーザのホスト一覧が提供されます。この一覧の鍵を表示できますが、[Reflection Secure Shell の設定] ダイアログボックスの **[ホスト鍵]**『40ページ 』タブにある

[グローバルホスト鍵] 一覧で編集できません。

[ホスト鍵] タブ ([Reflection Secure Shell の設定])

表示方法『15ページ 』

[ホスト鍵] タブを使用してクライアントセッションに対するホストを認証する鍵を管理します。このタブを使用して、信頼されているホスト一覧の表示、ホスト鍵の追加や削除、および Reflection の不明ホストの処理方法の指定を行うことができます。

ホスト認証を使用すると、Secure Shell クライアントは Secure Shell サーバの識別情報を確認し、その情報を信頼できます。この認証は公開鍵認証を使用して実行されます。ホスト公開鍵がクライアントにまだインストールされていない場合は、初回の接続試行時に、これが不明なホストであることを示すメッセージが表示されます。このメッセージには、ホストを識別する指紋が含まれます。本当にユーザのホストであるかどうかを確認するには、正しい指紋であることを確認できるホストシステム管理者に問い合わせる必要があります。本当にユーザのホストであることが確認されるまでは、別のサーバがユーザのホストを偽装する「中間者」攻撃に晒される危険性があります。 この要求への応答で **[常時]** を選択すると、ホストが **[信頼されているホスト鍵]** 一覧に追加されます。ホスト管理者への連絡を不要にするため、最初に接続する前にホスト鍵を **[信頼されているホスト鍵]** 一覧に追加できます。

次のオプションがあります。

[厳格なホスト鍵の確認の履行]

不明ホストに接続する際に、Reflection が*ホスト鍵の確認*『38ページ 』を処理する方法を指定します。

[証明書よりも SSH 鍵を優先する]

Reflection の*ホスト鍵アルゴリズムの優先順位*『39ページ 』を指定します。この設定を選択しない (既定値にする) と、Reflection はホスト鍵の前にホスト証明書を要求します。この設定を選択すると、Reflection はホスト証明書の前にホスト鍵を要求します。

[信頼されているホスト鍵]

現在の Windows ユーザに信頼されているホストの一覧を表示します。**[インポート]** と **[削除]** を使用してこの一覧の内容を変更できます。

既定では、この一覧に含まれないホストに接続しようとすると、新しいホスト鍵を信頼するかどうかを確認されます。この要求への応答で **[常時]** を選択すると、ホストが **[信頼されているホスト鍵]** 一覧に追加されます。

[インポート]

ホストの公開鍵を **[信頼されているホスト鍵]** 一覧に追加します。

[削除]

選択した鍵を [信頼されているホスト鍵] 一覧から削除します。

警告:確認プロンプトは表示されません。また、この操作を取り消すことはできません。

[グローバルホスト鍵]

コンピュータのすべてのユーザが使用できる信頼されているホスト鍵の一覧を表示します。この一覧の項目は、表示はできますが編集はできません。

システム管理者は、グローバルの*既知のホストファイル*『39ページ 』を使用して **[グローバルホスト鍵]** 一覧を変更できます。

[ホスト鍵の信頼性] ダイアログボックス

この確認ダイアログボックスは、接続しているホストが信頼するホストではない場合に表示されます。この新しいホスト鍵を信頼して、接続を継続しますか?

ホスト認証を使用すると、Secure Shell クライアントは Secure Shell サーバの識別情報を確認し、その情報を信頼できます。この認証は公開鍵認証を使用して実行されます。ホスト公開鍵がクライアントにまだインストールされていない場合は、初回の接続試行時に、これが不明なホストであることを示すメッセージが表示されます。このメッセージには、ホストを識別する指紋が含まれます。本当にユーザのホストであるかどうかを確認するには、正しい指紋であることを確認できるホストシステム管理者に問い合わせる必要があります。本当にユーザのホストであることが確認されるまでは、別のサーバがユーザのホストを偽装する「中間者」攻撃に晒される危険性があります。

次のオプションがあります。

[常時]	接続を行って、信頼するホストの一覧にこのホストを追加します。信頼するホストの一覧からこのホストを削除するか、ホスト鍵が変更されないかぎり、以降、同じホストに接続する際に上記のメッセージが表示されることはありません。
[今回のみ]	接続を行いますが、信頼するホストの一覧にこのホストを追加しません。次に同じホストに接続する時は上記のメッセージが表示されます。
[いいえ]	接続を行わず、信頼するホストの一覧にこのホストを追加しません。

7 章

証明書の認証（PKI）

この章の内容

PKI と証明書

PKI (Public Key Infrastructure) は、電子証明書を使用して安全に通信できるようにするためのシステムです。Reflection for Secure IT では、ホストとユーザの両方の認証に PKI を使用できます。

公開鍵認証のように、証明書認証では公開/秘密鍵のペアを使用してホストの身元を確認します。ただし、証明書認証を使用すると、公開鍵が *電子証明書* 『147ページ 』内に含まれ、この場合 2 つの鍵のペアが使用されます。例えばサーバ認証の場合、ホストは秘密鍵を 1 つ保有し、CA がもう 1 つの鍵を保有します。ホストは、CA から証明書を取得します。この証明書には、ホストに関する識別情報、ホスト公開鍵のコピー、CA の秘密鍵を使用して作成された *電子署名* 『146ページ 』が含まれています。この証明書は、認証処理時にクライアントに送信されます。ホストから送信される情報の整合性を検証するには、クライアントが CA ルート証明書に含まれる CA の公開鍵のコピーを保有している必要があります。クライアントが、ホスト公開鍵のコピーを保有する必要はありません。

証明書認証により、公開鍵認証により発生するいくつかの問題が解決されます。例えばホスト公開鍵認証の場合、システム管理者はすべてのサーバのホスト鍵を各クライアントの既知のホスト格納場所に配布したり、クライアントユーザが既知のホストに接続する場合に、クライアントユーザに依頼してホストの身元を正しく確認する必要があります。証明書をホスト認証に使用すると、単一の CA ルート証明書を使用して複数のホストを認証できます。多くの場合、必要な証明書は Windows の証明書格納場所ですでに使用可能です。

同様に、公開鍵をクライアント認証に使用すると、各クライアント公開鍵をサーバにアップロードし、その鍵を認識するようにサーバを構成する必要があります。証明書認証を使用すると、単一の CA ルート証明書を使用して複数のクライアントユーザを認証できます。

電子証明書の格納場所

電子証明書は、コンピュータの証明書格納場所に保存されています。証明書格納場所には、相手の身元を確認するのに使用する証明書が含まれています。また、自分の身元を相手に示すのに使用する個人用証明書が含まれていることもあります。個人用証明書は、コンピュータにある秘密鍵に関連付けられています。

Reflection は、次のどちらかまたは両方の格納場所にある電子署名を使用するように構成できます。

- **Windows の証明書格納場所**

  この格納場所は、Reflection、Web ブラウザ、メールクライアントなど、複数のアプリケーションで使用できます。この格納場所の証明書のいくつかは、Windows オペレーティングシステムのインストール時にインストールされます。また、インターネットサイトに接続して信頼関係を確立したり、ソフトウェアをインストールしたり、暗号化された電子メールや電子署名のある電子メールを受け取った時にも追加されます。Windows の格納場所に証明書を手作業でインポートすることもできます。この格納場所の証明書は、Windows の証明書マネージャを使用して管理します。

- **Reflection の証明書格納場所**

  この格納場所は、Reflection アプリケーションでのみ使用されます。この格納場所に証明書を追加するには、証明書を手作業でインポートする必要があります。証明書はファイルからインポートでき、スマートカードなどのハードウェアトークン上の証明書を使用することもできます。この格納場所の証明書は、*Reflection 証明書マネージャ*『49ページ 』を使用して管理します。

Reflection のアプリケーションは、Reflection の格納場所にある証明書のみ、または Windows の格納場所と Reflection の格納場所の両方を使用して認証するように構成することができます。Windows の証明書格納場所にある証明書を使用したホスト認証を有効にすると、既存の証明書を使用して認証できるので、証明書をインポートしなくて済むことがあります。Windows の証明書格納場所にある証明書を使用した認証を無効にすると、認証で使用する証明書を詳細に制御できるようになります。詳細については、「*Windows の証明書格納場所を使用した認証の有効化と無効化*『45ページ 』」を参照してください。

証明書を使用したクライアント認証の構成

電子証明書『147ページ 』は、Secure Shell クライアントセッションでのホスト認証とクライアント*認証*『147ページ 』の両方またはそのいずれかで使用できます。証明書は必須ではなく、既定では使用されません。このトピックでは、証明書を使用して認証されるように Reflection for Secure IT クライアントを構成する方法について説明します。Secure Shell サーバの構成方法については、サーバのマニュアルを参照してください。

クライアントに証明書認証を構成するには

1 個人用証明書および関連する秘密鍵を含むファイル（*.pfx または *.p12 ファイル）を入手します（証明書は、認証局から入手できます）。

2 このファイルを使用して、証明書を *Reflection 証明書マネージャ*『49ページ 』または Windows の証明書格納場所の [個人] タブにインポートします。

3 Reflection for Secure IT で、**[Reflection Secure Shell の設定]**『15ページ 』ダイアログボックスを開きます。

4 **[全般]** タブで、**[ユーザ認証]** の **[公開鍵]** が選択されていること（既定）を確認します。

5 **[ユーザ鍵]** タブで、使用可能な鍵の一覧から使用したい証明書を検索し、使用できるようにするために **[使用]** 列のチェックボックスをオンにします。

証明書を使用したサーバ認証の構成

証明書を使用してホストを認証するために Reflection クライアント側で必要な構成は、サーバが提示する証明書の認証に必要な CA 証明書をインストールすることだけです。証明書をインストールする要件は、Reflection がどのように構成されていて、証明書がどのように作成されたかによって異なります。

- 証明書が VeriSign や Thawte などのよく知られた認証局 (CA) から取得されたもので、Windows の証明書格納場所を使用してホスト証明書に対応するように Reflection を構成した場合は、使用しているコンピュータに証明書をインストールしなくて済むことがあります。発行者を信頼された CA として識別する証明書が、使用しているシステムの信頼されたルート認証局の一覧にすでに含まれている場合があります。
- Reflection の格納場所を使用して認証を要求するように Reflection を構成した場合、各クライアントコンピュータは必要な CA 証明書を Reflection の格納場所にインポートする必要があります。
- 会社独自の認証局を作成した場合、各クライアントコンピュータはその認証局のルート証明書をインポートする必要があります。構成に応じて、Windows の証明書格納場所または Reflection の証明書格納場所にインポートします。

Windows の証明書格納場所の使用の有効化と無効化

Reflection の Secure Shell および SSL/TLS セッションは、ホストおよびユーザ認証用の *電子証明書* 『147ページ 』の使用に対応しています。Reflection のアプリケーションは、Reflection の格納場所にある証明書のみ、または Windows の格納場所と Reflection の格納場所の両方を使用して認証するように構成することができます。

ホスト認証

Windows の証明書格納場所が使用できるようにすると、ホスト認証に使用する証明書のインポートが不要になります。ホスト証明書を VeriSign や Thawte などの有名な *認証極* 『143ページ 』(CA) から取得した場合、発行者を信頼された CA として識別する証明書が、システムの信頼されたルート認証局の一覧にすでに含まれている場合があります。システムの格納場所が使用できる場合、Reflection クライアントは Reflection とシステムの両方の格納場所で証明書を探します。

Windows の証明書格納場所を使用できないようにすると、認証で使用する証明書を詳細に制御できるようになります。Windows の格納場所には、さまざまな方法で証明書を追加できます。Reflection セッションの認証にこれらの証明書を全部は使用したくない場合もあるかもしれません。Windows の格納場所を使用できないようにすると、Reflection の格納場所にインポートした証明書のみがホスト認証に使用されます。

Windows の格納場所にある証明書を使用したホスト認証を有効 (または無効) にするには

1 *[Reflection 証明書マネージャ]* 『49ページ 』を開きます。

2 **[信頼された認証局]** 『50ページ 』タブをクリックします。

3 **[システムの格納場所にある証明書を使用して SSH に接続する]** と **[システムの格納場所にある証明書を使用して SSL/TLS に接続する]** の両方または一方をオン (またはオフ) にします。

ユーザ認証

Reflection は、Windows の格納場所と Reflection の格納場所にある個人証明書を同じ方法で使用します。使用可能な個人証明書には、 Windows の個人用格納場所にある証明書、*Reflection の個人用格納場所*『49ページ 』にある証明書、および*構成済みのハードウェアトークン*『53ページ 』(スマートカードなど) にある証明書が含まれます。

- Reflection Secure Shell セッションを構成済みの場合は、[Secure Shell 設定] ダイアログボックスの **[ユーザ鍵]** タブから、ユーザ認証に使用する証明書を指定する必要があります。
- Reflection SSL/TLS セッションを構成済みの場合は、いずれかの格納場所にあるすべての証明書をユーザ認証に自動的に使用できます。

証明書取り消しの確認の構成

Reflection SSL/TLS 接続と Secure Shell 接続では、*電子証明書*『147ページ 』を使用したホスト認証を構成できます。失効していない証明書を確実に使用するには、*CRL*『143ページ 』または OCSP レスポンダを使用して証明書の取り消しを確認するように Reflection を構成します。

Reflection で CRL の確認が有効になっている場合は、証明書の CRL Distribution Point (CDP) フィールドに指定されているすべての場所で CRL が必ず確認されます。また、LDAP ディレクトリにある CRL を確認したり、OCSP レスポンダを使用するように Reflection を構成することもできます。

Reflection では、証明書取り消しの確認の既定値は現在のシステム設定に基づいて決まります。システムが CRL の確認を行うように構成されている場合は、既定ですべての Reflection セッションにおいて CRL を使用して証明書取り消しが確認されます。

注意:Reflection が *DOD PKI モード*『138ページ 』で実行されている場合、証明書取り消しは常に有効であり、無効にすることはできません。

すべての SSH セッションで CRL の確認を有効にするには

1 Internet Explorer で **[ツール]** - **[インターネットオプション]** - **[詳細設定]** を選択します。

2 **[セキュリティ]** の下の **[サーバー証明書の取り消しを確認する]** をオンにします。

Reflection では、CRL または OCSP レスポンダを使用した証明書取り消し確認を行えます。

Secure Shell セッションでの CRL の確認を有効にするには

1 **[Reflection Secure Shell の設定]** ダイアログボックスを開きます。

2 **[PKI]** タブをクリックします。

3 **[OCSP を使用する]** または **[CRL を使用する]** をオンにします。

SSL/TLS セッションでの CRL の確認を有効にするには

1 **[セキュリティのプロパティ]** ダイアログボックスを開きます。

2 **[SSL/TLS]** タブで [PKI の構成] をクリックします (**[SSL/TLS セキュリティを使用する]** がオンになっている必要があります)。

3 **[OCSP を使用する]** または **[CRL を使用する]** をオンにします。

注意：証明書で必要な CRL レスポンダおよび OCSP レスポンダは、証明書の AIA 拡張および CDP 拡張に指定されます。この情報が証明書で提供されない場合、[Reflection 証明書マネージャ] の **[OCSP]**『53ページ 』タブおよび **[LDAP]**『52ページ 』タブを使用して構成できます。

LDAP ディレクトリを使用した中間証明書の配布

Reflection SSL/TLS 接続と Secure Shell 接続では、*電子証明書*『147ページ 』を使用したホスト認証を構成できます。*Reflection 証明書マネージャ*『45ページ 』の構成方法に応じて、Reflection の格納場所にある証明書のみ、または Windows および Reflection の両方の格納場所にある証明書が Reflection で使用されます。Windows の格納場所には、中間証明書と信頼されたルート証明書が保存されます。Reflection の格納場所には、信頼されたルート証明書のみが保存されます。また、LDAP サーバから中間証明書を検索するように Reflection を構成することもできます。

LDAP ディレクトリに保存されている中間証明書を検索するように Reflection を構成するには、[Reflection 証明書マネージャ] の **[LDAP]**『52ページ 』タブ で LDAP サーバ（1 台または複数）を指定します。

LDAP サーバの構成

Reflection で LDAP ディレクトリ内の証明書を検索できるのは、LDAP 識別名（DN）が証明書の件名フィールドの内容と完全に一致する場合のみです。例えば、証明書の件名フィールドに以下のオブジェクトが表示されるとします。

- CN = Some CA
- O = Acme
- C = US

この場合、LDAP ディレクトリのエントリの DN は「CN = Some CA, O=Acme, C = US」である必要があります。

この DN で識別される LDAP エントリの属性は、以下のいずれかを含む必要があります（Reflection ではこれらの属性を上から下に検索します）。

属性	OID (Object Identifier - オブジェクト識別子)
userCertificate;binary	2.5.4.36
cACertificate;binary	2.5.4.37
userCertificate	2.5.4.36
cACertificate	2.5.4.37
mosaicKMandSigCertificate	2.16.840.1.101.2.1.5.5
sdnsKMandSigCertificate	2.16.840.1.101.2.1.5.3
fortezzaKMandSigCertificate	2.16.840.1.101.2.1.5.5
crossCertificatePair;binary	2.5.4.40

crossCertificatePair 2.5.4.40

[PKI] タブ ([Reflection Secure Shell の設定] ダイアログボックス)

表示方法『15ページ 』

Reflection Secure Shell セッションでの PKI 設定を構成するには、このタブを使用します。

次のオプションがあります。

[証明書のホスト名と対象ホスト名が一致するかどうかを確認する] ホスト証明書の検証時にホスト名が一致していなければならないかどうかを指定します。この設定がオンになっている場合 (既定値)、Reflection で構成したホスト名が、証明書の **CommonName** フィールドまたは **SubjectAltName** フィールドに入力されているホスト名に一致していなければなりません。

[OCSP を使用] Reflection が、ホスト証明書の検証時に OCSP (Online Certificate Status Protocol) レスポンダを使用して証明書の破棄を確認するかどうかを指定します。OCSP レスポンダは証明書自体の AIA 拡張で指定できます。また、Reflection 証明書マネージャの **[OCSP]**『53ページ 』タブを使用して OCSP レスポンダを指定することもできます。

[CRL を使用] Reflection が、ホスト証明書の検証時に *CRL*『143ページ 』(Certificate Revocation List) を使用して証明書の破棄を確認するかどうかを指定します。CRL は証明書自体の CDP 拡張で指定できます。また、Reflection 証明書マネージャの **[LDAP]**『52ページ 』タブを使用して CRL を指定することもできます。

注意: この設定の既定値は、CRL の確認に関する現在のシステム設定に従います。システム設定を表示して編集するには、Internet Explorer を起動して、**[ツール]** - **[インターネットオプション]** - **[詳細設定]** コマンドに進みます。**[セキュリティ]** の下の **[サーバー証明書の取り消しを確認する]** を探します。

[Reflection 証明書マネージャ] **Reflection 証明書マネージャ**を開きます。これを使用して、Reflection の格納場所にある証明書を管理したり、PKI 設定を指定したりできます。

[システム証明書の表示] **Windows 証明書マネージャ**を開きます。これを使用して、システムの格納場所にある証明書を管理できます。

注意

- このダイアログボックスで構成した設定は、*Secure Shell 構成ファイル*『19ページ 』に保存されます。また、任意のテキストエディタを使用して、このファイルを編集して Secure Shell 設定を構成することもできます。
- この構成ファイルの内の設定は、現在指定されている *SSH 構成セクション*『116ページ 』用に保存されます。

[Reflection 証明書マネージャ]

表示方法『49ページ 』

Reflection アプリケーションは、Windows の証明書格納場所または Reflection の証明書格納場所（あるいはその両方）にある*電子証明書*『147ページ 』を使用して認証できます。Reflection の証明書格納場所は、Secure Shell セッションまたは SSL/TLS セッションでの認証で使用することができます。

Reflection 証明書マネージャを使用して、Reflection の格納場所にある電子証明書を管理したり、Reflection の PKI 対応のほかのオプションを構成します。

Reflection 証明書マネージャを開く

この手順を使用して Reflection 証明書マネージャを起動します。

Reflection 証明書マネージャを開くには

1 **[Reflection Secure Shell の設定]**『15ページ 』ダイアログボックスを開きます。

2 **[PKI]** タブで、**[Reflection 証明書マネージャ]**をクリックします。

[個人] タブ ([Reflection 証明書マネージャ])

表示方法『49ページ 』

このタブは、Reflection の証明書格納場所にある個人用*証明書*『147ページ 』の管理に使用します。個人用証明書はユーザ（クライアント）認証に使用されます。

次のオプションがあります。

[インポート]
Reflection の格納場所に証明書を追加します。インポートしたファイル（通常、*.pfx または *.p12）には秘密鍵が含まれている必要があります。ファイルがどのように作成されたかによっては、ファイルをインポートする前にパスワードの入力が要求されることがあります。

Reflection の格納場所にある秘密鍵を保護できるように、*パスフレーズ*『146ページ 』の入力が要求されます。パスフレーズを指定すると、この証明書を使用してホストに認証する時に、このパスフレーズを入力するように要求されます。

[削除]
選択した証明書を Reflection の格納場所から削除します。

[表示]
選択した証明書を表示します。

[パスフレーズの変更]
選択した証明書の*パスフレーズ*『146ページ 』を変更します。

[信頼された認証局] タブ ([Reflection 証明書マネージャ])

表示方法『49ページ 』

このタブは、信頼された認証局から提供された、Reflection の格納場所にある証明書の管理に使用します。Reflection は、信頼された認証局の格納場所にあるあらゆる証明書をホスト (サーバ) の認証に使用します。

[インポート]

Reflection の格納場所に証明書 (通常、*.cer または *.crt) を追加します。

[削除]

選択した証明書を Reflection の格納場所から削除します。

[表示]

選択した証明書を表示します。

[共通のアプリケーションデータフォルダに信頼された証明書を格納]

[インポート] ボタンを使用して追加する信頼されたルートは、既定では、次の Reflection 格納場所に保存されるため、現在のユーザアカウントのみが使用できます。

個人用ドキュメントフォルダ
¥Attachmate¥Reflection¥.pki¥trust_store.p12

[共通のアプリケーションデータフォルダに信頼された証明書を格納] をオンにすると、証明書が次の場所にインポートされるため、コンピュータ上のすべてのユーザが使用できます。

共通アプリケーションデータフォルダ『145ページ 』
¥Attachmate¥Reflection¥.pki¥trust_store.p12

注意:

- この設定の値は保存されません。この設定をオンにするかオフにするかは、ダイアログボックスが開いている間にどの証明書格納場所を表示し、編集するかにのみ影響します。共有格納場所がある場合、既定では、ダイアログボックスを開くとこの設定が選択されます。共有格納場所がない場合、既定ではこの設定は選択されません。
- 共有格納場所がある場合、信頼されたルートは共有格納場所のみから読み取られます。個々のユーザアカウント用に構成した信頼されたルートは影響を及ぼさなくなります。
- 共有格納場所を作成した後で、ユーザ固有の信頼されたルート格納場所に戻すには、共有 trust_store.p12 ファイルを削除するか、名前変更する必要があります。この設定をオフにするだけだと、続く変更によって個人の格納場所は変更されますが、trust_store.p12 が共通アプリケーションデータフォルダにある限り、個人の格納場所は Reflection の動作に影響を及ぼしません。
- オペレーティングシステムが、common_application_data_folder『145ページ 』¥Attachmate¥Reflection へのユーザ書き込みアクセス権を拒否するように管理者によって構成されている場合、書き込みアクセス許可を拒否されたユーザはこの設定を使用できず、信頼されたルートの共有格納場所のアイテムを変更することはできません。

[システムの格納場所にある証明書を使用して SSH に接続]

この項目をオンにすると、Reflection は SecureShell 接続を確立する際に、Windows の格納場所にある証明書を使用してホストを認証します。

Reflection アプリケーションが、Reflection の格納場所にある証明書だけを使用してホストを認証するようにするには、この設定をオフにします。

[システムの格納場所にある証明書を使用して SSL/TLS に接続]

この項目をオンにすると、Reflection は SSL/TLS 接続を確立する際に、Windows の格納場所にある証明書や Reflection の格納場所にインポートした証明書を使用してホストを認証します。

Reflection アプリケーションが、Reflection の格納場所にある証明書だけを使用してホストを認証するようにするには、この設定をオフにします。

[MD5 署名済証明書を許可]

[MD2 署名済証明書を許可]

これらの項目を選択すると、Reflection は、指定ハッシュで署名された中間 CA 証明書を受け付けます。これらの項目を選択しないと、中間証明書が指定ハッシュで署名されている場合、証明書の検査が失敗します。

- これらの証明書ハッシュ設定は、中間 CA 証明書のみに影響します。Reflection は、署名のハッシュの種類に関係なく、信頼されたルート格納場所に追加された任意の証明書を受け付けます。
- Reflection がグループポリシーによって DOD PKI モードで動作するように構成されている場合、これらの設定は使用できません。

[LDAP] タブ ([Reflection 証明書マネージャ])

表示方法『49ページ 』

Lightweight Directory Access Protocol (LDAP) は、一元化された場所に情報を格納し、その情報をユーザに配信するために使用できる標準プロトコルです。 管理者は、証明書で認証中のユーザが必要とする情報を配信するように LDAP サーバを構成できます。これには、次の情報が含まれます。

- 証明書の失効一覧 (CRL) - 使用されている証明書が認証局によって失効されていないこと保証するために使用します。
- 中間証明書 - サーバ証明書から信頼されたルート認証局への有効な証明書経路を確立するのに必要です。

Reflection 証明書マネージャの [LDAP] タブを使用して、この情報を配信する LDAP サーバを一覧表示できます。 次のオプションがあります。

[追加]	LDAP サーバを一覧に追加します。次の URL 形式を使用してサーバを指定します。
	`ldap://hostname[:portnumber]`
	例えば、次のように入力します。
	`ldap://ldapserver.myhost.com:389`

[変更]	サーバ URL を編集します。
[削除]	選択したサーバを一覧から削除します。

LDAP ディレクトリの構成

Reflection が LDAP ディレクトリに格納された情報を処理する方法の詳細については、次のリンクを参照してください。

- *CRL*『53ページ 』
- *中間証明書*『47ページ 』

注意

- CRL の確認を使用するように LDAP サーバを構成する必要はありません。Reflection で CRL の確認が有効になっている場合は、証明書の CRL Distribution Point (CDP) フィールドに指定されているすべての場所で CRL が必ず確認されます。LDAP サーバを構成すると、CRL 一覧を検索する追加のメカニズムが提供されます。
- Reflection は、SSL を使用して LDAP データを転送するために LDAPS スキーム (例えば、ldaps://hostname:port) を使用しているサーバ URL をサポートしていません。

CRL の確認を使用するように LDAP サーバを構成する

Reflection は、LDAP 識別名 (DN) が CRL の発行者フィールドの内容と正確に一致する場合にだけ、LDAP ディレクトリ内で CRL を検索できます。例えば、CRL の発行者フィールドに次のオブジェクトが表示されている場合:

- CN = Some CA
- O = Acme
- C = US

LDAP ディレクトリ内のエントリの DN は次に正確に一致している必要があります。"CN = Some CA, O=Acme, C = US"

この DN によって識別される LDAP エントリの属性には、次のいずれかが含まれている必要があります(Reflection は、これらの属性を上から下へ順番に検索します)。

属性	OID (オブジェクト識別子)
certificateRevocationList;binary	2.5.4.39
authorityRevocationList;binary	2.5.4.38
certificateRevocationList	2.5.4.39
authorityRevocationList	2.5.4.38
deltaRevocationList;binary	2.5.4.53
deltaRevocationList	2.5.4.53
mosaicCertificateRevocationList	2.16.840.1.101.2.1.5.45
sdnsCertificateRevocationList	2.16.840.1.101.2.1.5.44
fortezzaCertificateRevocationList	2.16.840.1.101.2.1.5.45

[OCSP] タブ ([Reflection 証明書マネージャ])

表示方法『49ページ 』

証明書の破棄確認用に 1 つまたは複数の OCSP レスポンダを構成します。一覧にサーバを追加するには、**[追加]** をクリックします。既定では、追加した各サーバには、一覧の先頭のサーバから順次、証明書の有効性の問い合わせが実施されます。サーバの横のチェックボックスをオフにすると、サーバを一覧から削除することなく、そのサーバの証明書の確認は無効にされます。

[追加] OCSP サーバを一覧に追加します。次の URL 形式を使用してサーバを指定します。

URL:portnumber

例えば、次のように入力します。

www.ocspresponder.com:80

[変更] サーバ URL を編集します。

[削除] 選択したサーバを一覧から削除します。

[PKCS#11] タブ ([Reflection 証明書マネージャ])

表示方法『49ページ 』

[PKCS#11] タブを使用して、スマートカードや USB トークンのようなハードウェアデバイスを使用する認証を構成できます。ハードウェアデバイスは、PKCS#11 仕様に準拠している必要があります。

このタブには、現在使用可能なすべてのデバイスと、それらのデバイスの証明書または公開鍵が表示されます。提供されたチェックボックスを使用してデバイスの使用が有効になっている場合、Reflection は自動的に、デバイスの証明書または鍵を使用してユーザ認証を行います。

ハードウェアトークンを認証するように Reflection を構成する前に、トークンプロバイダによって提供されたソフトウェアをインストールする必要があります。トークンを使用して認証を構成するには、ハードウェアデバイスにアクセスするためにプロバイダが使用しているライブラリファイル (.dll) の名前と場所も把握しておく必要があります。

次のオプションがあります。

[プロバイダ] 一覧 現在使用可能なデバイスを表示します。一覧表示されたデバイスの認証を無効にするには、チェックボックスをオフにします。

[デバイスの内容] 選択されたデバイスで使用可能な鍵と証明書を表示します。

[証明書の表示] 選択した証明書を表示します。

[トークンが削除された場合は自動的に切断する] オンの場合、接続はトークンが存在する間だけアクティブになります。

[切断待ち時間(秒)] トークンが削除された後、切断されるまでの待ち時間を秒数で指定します。

[PKCS#11 プロバイダ] ダイアログボックス

表示方法『49ページ 』

次のオプションがあります。

[プロバイダの DLL] ハードウェアデバイスにアクセスするために使用されているライブラリのファイル名と場所を指定します。これは、通常、Windows システムフォルダにインストールされています。正しいファイルを確認するためにデバイスの提供元に問い合わせる必要が生じることがあります。

[スロット ID] 認証に使用されているカードを保持するカードスロットを識別します。

[追加パラメータ] ハードウェアデバイス上の情報にアクセスするために必要な追加パラメータを指定します。

8 章

Secure Shell セッションの GSSAPI (Kerberos) 認証

この章の内容

GSSAPI 認証に対する Reflection Kerberos の使用

この手順では、Secure Shell サーバに対する認証に Windows のドメイン資格情報を使用するように Reflection Kerberos を構成します。 Secure Shell 接続の Kerberos 認証は、Secure Shell 構成ファイルに現在指定されている SSH 構成セクションで有効になります。

システム管理者が Kerberos 構成ファイルを PC にインストールしている場合、Reflection アプリケーションの初回起動時に、Reflection Kerberos が自動的に構成されます。 Reflection Kerberos 設定はユーザ単位でレジストリに保存され、Kerberos クライアントを使用するすべての Reflection アプリケーションで使用できます。

GSSAPI 認証に Reflection Kerberos を使用するには

1 **[Reflection Secure Shell の設定]**『15ページ 』ダイアログボックスを開きます。

2 **[全般]** タブの **[ユーザ認証]** で、**[GSSAPI/Kerberos]** チェックボックスをオンにします。

3 **[GSSAPI]** タブで、**[Reflection Kerberos]** をオンにします。

4 **[構成]** をクリックします。

5 **[Reflection Kerberos 初期構成]** ダイアログボックスで、プリンシパル名、レルム、KDC ホストを入力します。システムに Kerberos がすでに構成されている場合、代わりに Reflection Kerberos マネージャが起動します。

注意: 認証後、Reflection Kerberos は Kerberos 発券許可チケット (TGT) をホストに転送します。 チケット転送を無効にするには、「*Secure Shell セッションにおける Kerberos チケット転送*『55ページ 』」を参照してください。

Secure Shell セッションにおける Kerberos チケット転送

既定で、認証後 Reflectionは Kerberos発券許可チケット (TGT)をホストに転送します。

以下のいずれかの方法で、チケット転送を無効にできます。

- **[Reflection Secure Shell の設定]** ダイアログボックスの **[GSSAPI]**『56ページ 』タブの **[資格情報を委任する]** 設定をオフにします。この設定は、Secure Shellプロトコルバージョン 2 接続に影響します。

- *Secure Shell 構成ファイル*『19ページ 』を編集します。使用するプロトコルに応じて、以下の行の一方または両方を使用します。最初の行ではプロトコルバージョン 1 のチケット転送を無効にし、2 番目の行ではプロトコルバージョン 2 のチケット転送を無効にします。

  ```
  KerberosTgtPassing no
  ```

  ```
  GssapiDelegateCredentials no
  ```

- プリンシパルプロファイルで使用されるレルムのチケット転送を無効にするには、Reflection Kerberos マネージャを使用します (システムで使用可能な場合)。これらの変更は Reflection Kerberos を使用するように構成された Secure Shell セッションに影響しますが、SSPI を使用するように構成されたセッションには影響しません。上記の方法のいずれかを使用してチケット転送を構成している場合、Reflection Kerberos マネージャで行われた変更は無視されます。

GSSAPI Secure Shell セッションのサービスプリンシパルの指定

サービスプリンシパル名とは、Reflection が Kerberos Key Distribution Center (KDC) へサービスチケット要求を送信する時に使用する名前です。形式は次のとおりです。

```
hostname.domain.com@REALM
```

Reflection によって使用される名前は、[Reflection Secure Shell の設定] ダイアログボックスの **[GSSAPI]**『56ページ 』タブで構成する設定によって決まります。**[既定のサービスプリンシパル名を使用する]** がオン (既定) の場合、ホスト名の値は接続先の Secure Shell サーバの名前で、レルムの値は選択した GSSAPI プロバイダによって決まります。

- *Reflection Kerberos*『55ページ 』を使用している場合、レルム名は、既定プリンシパルプロファイルで指定したものになります。
- SSPI を使用している場合、レルム名は Windows のドメイン名です。

既定以外の値を指定するには、**[サービスプリンシパル]** 設定を使用します。GSSAPI プロバイダとして SSPI を選択している場合、この設定を使用して、Windows ドメインとは異なるレルムのサービスプリンシパルを指定できます。次のように、完全なホスト名の後に @、そしてレルム名を続けてください。

```
myhost.myrealm.com@MYREALM.COM
```

[GSSAPI] タブ ([Reflection Secure Shell の設定] ダイアログボックス)

表示方法『15ページ 』

[Reflection Secure Shell の設定] ダイアログボックスの **[GSSAPI]** タブを使用して、GSSAPI 認証の設定を指定できます。このタブの項目は、**[一般]**『15ページ 』タブの **[ユーザ認証]** 一覧で **[GSSAPI/Kerberos]** が選択されている場合にだけ使用可能です。

次のオプションがあります。

[SSPI] — Microsoft SSPI (Security Services Provider Interface) を使用します。SSPI では、Windows ドメインのログイン資格情報を使用して、Secure Shell サーバに対して認証します。これをオンにすると、Reflection Kerberos クライアントを構成する必要がなくなるため、セットアップが簡素化されます。

[Reflection Kerberos]	Kerberos/GSSAPI 認証のために Reflection Kerberos クライアントを使用します。Reflection Kerberos クライアントを使用して接続する前に、使用するコンピュータで Reflection Kerberos を構成する必要があります。
[構成]	Reflection Kerberos クライアントを構成します。このボタンは、**[Reflection Kerberos]** が選択された GSSAPI プロバイダである場合のみ使用できます。
[資格情報を委任]	GSSAPI がホストへ Kerberos 発券許可チケット (TGT) を転送するかどうかを指定します。
[既定のサービスプリンシパル名を使用]	Kerberos Key Distribution Center (KDC) へサービスチケット要求を送信する時に Reflection が使用する名前を指定します。ホスト名の値は、接続先の Secure Shell サーバの名前です。レルム値は、選択した GSSAPI プロバイダによって異なります。
[サービスプリンシパル]	既定以外のサービスプリンシパル値を指定します。

9 章

ポート転送

この章の内容

トンネリングとしても知られるポート転送は、アクティブなセッションにおける Secure Shell チャネルを通じて通信をリダイレクトするための方法を提供します。ポート転送が構成されると、指定のポートへ送信されるすべてのデータは、保護されたチャネルを通じてリダイレクトされます。ローカルポート転送またはリモートポート転送のいずれかを構成できます。「ローカル」および「リモート」という用語は、Secure Shell クライアントを基準にしてリダイレクトされたポート位置を示します。Reflection では、TCP 通信と FTP 通信の両方のローカルポート転送に対応しています。リモートポート転送は、TCP 通信のみに対応します。

用語集

ポート転送には、クライアントアプリケーションとサーバアプリケーションの 2 つのセット (Secure Shell クライアントとサーバ、およびデータが転送されるクライアント/サーバのペア) が必要です。このガイドでは、ポート転送に関連して以下のように定義された用語を使用します。

用語	定義
Secure Shell サーバ	Reflection for Secure IT サーバデーモン
Secure Shell サーバホスト	Secure Shell サーバを実行しているコンピュータ
Secure Shell クライアント	Reflection for Secure IT クライアントアプリケーション
Secure Shell クライアントホスト	Secure Shell クライアントを実行しているコンピュータ
アプリケーションクライアント	そのデータを転送したいクライアント/サーバのペアのクライアントプログラム。例えば、メールクライアントまたは Web ブラウザです。
アプリケーションクライアントホスト	アプリケーションクライアントを実行しているコンピュータ。通常は Secure Shell サーバホストまたは Secure Shell クライアントホストのいずれかになりますが、3 番目のホストにすることもできます。

用語	定義
アプリケーションサーバ	メールサーバまたは Web サーバなど、アプリケーションクライアントと通信するサーバプログラム
アプリケーションサーバホスト	サーバアプリケーションを実行しているコンピュータ。Secure Shell サーバホストまたは Secure Shell クライアントホストのいずれか、または 3 番目のホストにすることもできます。

ローカルポート転送

ローカルポート転送を使用して、Secure Shell クライアントと同じコンピュータ上で実行されているアプリケーションクライアントからデータを安全に転送できます。ローカルポート転送を構成する時は、データの転送に使用する任意のローカルポート、およびデータを受信する着信先ホストとポートを指定します。ローカルポート転送は次のように動作します。

1. Secure Shell 接続が確立されると、Secure Shell クライアントは、指定のローカルポートを使用して、ローカルコンピュータ (Secure Shell クライアントを実行しているコンピュータ) 上のリスニングソケットを開きます。 ほとんどの場合、Secure Shell クライアントホストを実行しているアプリケーションのみがこのソケットを使用できます。

   ゲートウェイポート設定は、ローカルに転送されるポートをリモートアプリケーションが使用できるかどうかを制御します。既定ではこの設定は無効で、クライアントは、ローカルポート転送のためにソケットを開いた時に、ループバックアドレス (「localhost」または 127.0.0.1) を使用します。これにより、他のコンピュータで実行中のアプリケーションは、転送されるポートに接続できなくなります。ゲートウェイポートを有効にすると、リモートアプリケーションクライアントは、Secure Shell クライアントの Ethernet アドレス (IP アドレス、URL、DNS 名など) を使用してソケットを開くことができます。例えば、acme.com で実行中の Secure Shell クライアントがポート 8088 を転送するよう構成されているとします。ゲートウェイポートが無効な場合、転送されるソケットは localhost:8088 です。ゲートウェイポートが有効な場合、転送されるソケットは acme.com:8088 です。

注意: ゲートウェイポートを有効化すると、使用しているクライアントホスト、ネットワーク、接続のセキュリティの低下を招きます。この理由は、リモートアプリケーションが、認証なしで、システム上の転送されたポートを使用することが可能になるからです。

2. アプリケーションクライアントは、(アプリケーションサーバホストおよびポートに直接接続するのでなく) 転送ポートに接続するように構成されます。 そのクライアントが接続を確立すると、すべてのデータがリスニングポートに送信され、Secure Shell クライアントにリダイレクトされます。
3. Secure Shell クライアントはデータを暗号化し、Secure Shell チャネルを通じて Secure Shell サーバに安全にデータを送信します。
4. Secure Shell サーバはデータを受信して解読し、アプリケーションサーバによって使用される送信先ホストおよびポートにリダイレクトします。

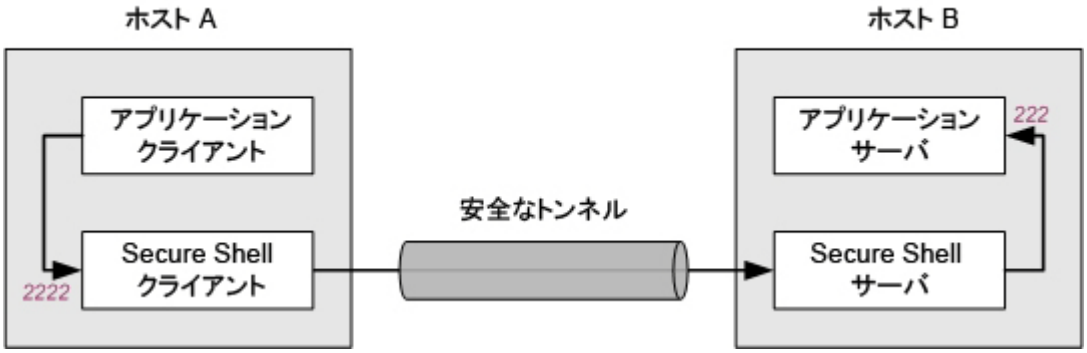
注意: 最終宛先ホストおよびポートが Secure Shell サーバホスト上にない場合、Secure Shell ホストとアプリケーションサーバホスト間でデータは平文で送信されます。

5. アプリケーションサーバからの戻りデータは Secure Shell サーバに送られ、Secure Shell サーバは戻りデータを暗号化し、SSH トンネルを通じて Secure Shell クライアントに安全に送信します。 Secure Shell クライアントはデータを解読し、元のアプリケーションクライアントにデータをリダイレクトします。

ローカルポート転送の一般的なコマンドライン構文は以下のとおりです。

```
ssh -L listening_port:app_host:hostport user@sshserver
```

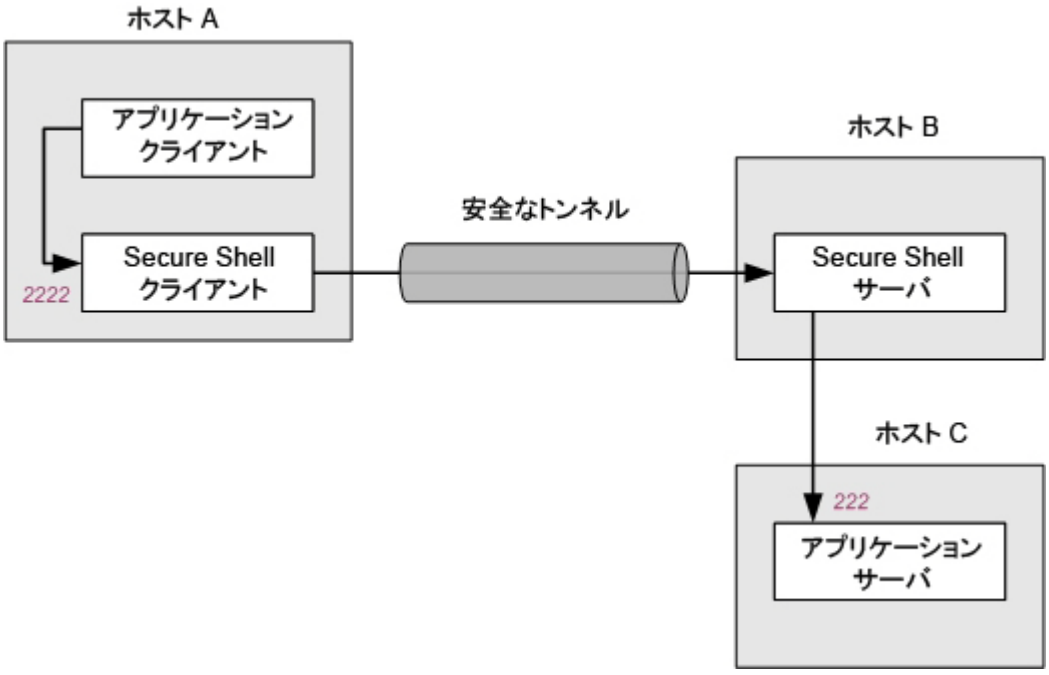
以下に示す図は、ローカルポート転送の 2 とおりの使用法を示しています。



上記の構成では、アプリケーションクライアントと Secure Shell クライアントの両方がホスト A で実行されます。Secure Shell サーバとアプリケーションサーバの両方はホスト B で実行されます。ホスト A のポート 2222 へ送信されたすべてのデータはホスト B のポート 222 へ転送されます。この配置では、転送中のすべてのデータが安全に暗号化されます。これは、以下のコマンド（localhost はホスト B のループバックアドレスを識別します）によって構成します。

```
ssh -L 2222:localhost:222 user@HostB
```

以下の図は、3 番目のホストへのローカルポート転送を示しています。この構成では、アプリケーションサーバが、Secure Shell サーバとは異なるホストで実行されます。ホスト A のポート 2222 へ送信されたすべてのデータはホスト C のポート 222 へ転送されます。



これは、以下のコマンドによって構成します。

```
ssh -L 2222:HostC:222 user@HostB
```

注意： ホスト B とホスト C 間で送信されるデータは暗号化されません。

リモートポート転送

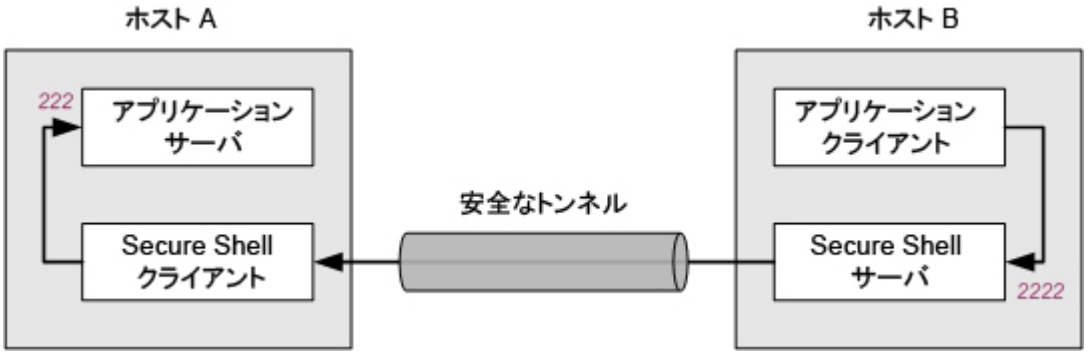
リモートポート転送を使用して、Secure Shell サーバホストで実行されているアプリケーションクライアントからデータを安全に転送できます。リモートポート転送を構成する時は、データの転送に使用する任意のリモートポート、およびデータを受信する着信先ホストとポートを指定します。リモートポート転送は次のように動作します。

1. Secure Shell 接続が確立されると、Secure Shell サーバは、指定したリスニングポートを使用して、Secure Shell サーバホスト上のリスニングソケットを開きます。
2. Secure Shell サーバホストで動作するクライアントアプリケーションは、(アプリケーションサーバホストおよびポートに直接接続する代わりに) リスニングポートに接続するように構成されます。そのクライアントが接続を確立すると、すべてのデータがリスニングポートに送信され、Secure Shell サーバにリダイレクトされます。
3. Secure Shell サーバはデータを暗号化し、SSH トンネルを通じて安全に Secure Shell クライアントにデータを送信します。
4. Secure Shell クライアントは、データを受信し、解読し、サーバアプリケーションが使用する (Secure Shell クライアントホスト上の) 宛先ホストおよびポートにリダイレクトします。
5. サーバアプリケーションからの戻りデータは Secure Shell クライアントに送られ、Secure Shell クライアントは戻りデータを暗号化し、SSH トンネルを通じて Secure Shell サーバに安全に送信します。Secure Shell サーバはデータを解読し、元のクライアントアプリケーションにデータをリダイレクトします。

リモートポート転送の場合の一般的なコマンドライン構文は次のとおりです。

```
ssh -R listening_port:app_host:hostport user@sshserver
```

次の図に、リモートポート転送構成の例を示します。



HostA では、アプリケーションサーバと Secure Shell クライアントが動作しています。HostB では、Secure Shell サーバとアプリケーションクライアントが動作しています。HostB のポート 2222 に送信されるすべてのデータが、HostA のポート 222 に転送されます。この配置では、通過中のすべてのデータが安全に暗号化されます。これを構成するコマンドは次のとおりです。

```
ssh -R 2222:localhost:222 user@HostB
```

TCP 通信の転送

この手順を使用して、アプリケーションクライアントとサーバ間でプレーンテキストで送信される TCP 通信内容を暗号化します（括弧で示した例では、Reflection for Secure IT を実行しているコンピュータの Web ブラウザとリモート Web サーバ間でデータを安全に送信するように Reflection for Secure IT クライアントを構成します）。

TCP 通信を転送するには

1 Reflection for Secure IT クライアントを開き、Secure Shell サーバホスト（例、MySSHserver.com）に接続するように構成します。

2 **[Reflection Secure Shell の設定]**『15ページ 』ダイアログボックスを開きます。**[トンネリング]** タブに移動します。

3 **[ポートのローカル転送]** の **[追加]** をクリックします。

4 **[ローカルポートの転送]** で、使用可能なローカルポートを指定します。通常、1024 より大きい値（例、8080）を入力できます。通常、1024 以下の値のポートはサービス用に予約されているため、使用できません。

5 **[転送先のホスト]** で、アプリケーションサーバホストの **[名前]**（例、WebServer.Acme.com）を指定します。

注意：このサーバホストが Secure Shell サーバホストと異なる場合、Secure Shell サーバと指定したサーバ間の通信は暗号化されません。指定したサーバが Secure Shell サーバホストと同じリモートコンピュータで実行されている場合、値 **localhost**（または IP 接続 127.0.0.1）を指定できます。この場合、すべての通信が暗号化されます。

---

6 **[ポート]** で、アプリケーションサーバで使用されるポート（例、Web サーバの場合 80 またはメールサーバの場合 110）を指定します。

注意：次の 2 つの手順は不要ですが、これらの手順を完了すると、Secure Shell トンネルの確立後、アプリケーションクライアントが自動的に起動されるように Reflection for Secure IT が構成されます。

---

7 （オプション）**[起動するアプリケーション]** で、トンネルを介して転送したいデータのクライアントアプリケーション名（例えば iexplore.exe）を指定します。システムパスにないアプリケーションの場合、完全なパス情報を含める必要があります。実行ファイルの検索に **[参照]** ボタンを使用して完全なパス情報を含めることができます。

8 （オプション）**[引数]** で、このアプリケーションに使用したいコマンドライン引数を指定します（例えば、http:¥¥localhost:8080 を使用してリダイレクトポート 8080 に接続するようにブラウザを設定できます）。アプリケーションクライアントを実行し、指定したポートに接続するよう構成する必要がある場合もあります。

9 **[OK]** をクリックして開いているダイアログボックスを閉じます。

注意：**[ローカルポート転送]** ダイアログボックスの **[OK]** ボタンは、すべての必須情報が入力されるまで使用できません。

---

10 Secure Shell ホストに接続します。

Secure Shell 接続の確立後、手順 7 で指定したアプリケーションが起動されます。転送されるローカルポート（この例では 8080）への接続が正しく構成されている場合、このポートからサーバアプリケーションにデータがリダイレクトされます。クライアントは、そのサーバに直接接続するよう構成されているかのように正確に実行されます。

FTP 通信の転送

Secure Shell のポート転送を使用して FTP プロトコル通信（FTP コマンドチャネルおよびすべてのデータチャネルを含む）を暗号化するには、次の手順に従います。 ポート転送を使用することで、FTP サーバに安全に接続でき、SFTP 接続では使用できないオプションやコマンドも含め、すべての FTP オプションおよびコマンドにアクセスできます。

注意：データチャネルの転送を有効にするためには、FTP クライアントは、パッシブ（PASV）モード（既定値）で通信するように構成されている必要があります。

FTP 通信を転送するには

1 FTP クライアントを起動します。

[FTP サイトに接続] ダイアログボックスが開きます（FTP クライアントが既に動作しており、このダイアログボックスが開かない場合は、**[接続]** - **[接続]** に進みます）。

2 次のいずれかのタスクを実行します。

結果	手順
新しいサイトの作成	**[FTP サイトに接続]** ダイアログボックスで **[新規]** をクリックします。 **[FTP サイトの追加]** ダイアログボックスで FTP サーバホストの名前または IP アドレスを入力し、**[次へ]** をクリックします。 **[ログイン情報]** ダイアログボックスで、**[ユーザ]** を選択します。
既存サイトの変更	**[FTP サイトに接続]** ダイアログボックスでサイトを選択します。

3 **[セキュリティ]** をクリックします。

4 **[Secure Shell]** タブをクリックします。

5 **[Reflection Secure Shell を使用する]** を選択します。

6 **[ポート転送を使用した FTP コマンドのトンネリング]** チェックボックスをオンにします。

7 この手順は、Secure Shell サーバが FTP サーバとは別のホスト上にある場合のみ、実行する必要があります。

- **[FTP ホストが Secure Shell ホストと異なる]** チェックボックスをオンにします。

注意：**[FTP ホストが Secure Shell ホストと異なる]** をオンにした場合、FTP コマンドとデータは、クライアントコンピュータから Secure Shell サーバに安全なトンネルを介して安全に送信されます。 コマンドとデータは、Secure Shell サーバと FTP サーバ間で暗号化されずに送信されます。

- **[SSH サーバアドレス]** で、Secure Shell サーバのホスト名または IP アドレスを入力します。
- **[SSH ユーザ名]** で、Secure Shell サーバでのログイン名を入力します。

8 次のいずれかのタスクを実行します。

目的	手順
新しいサイトの作成	**[OK]** をクリックして、**[セキュリティのプロパティ]** ダイアログボックスを閉じ、**[次へ]** をクリックします。 **[FTP ユーザログイン]** ダイアログボックスで、FTP サーバでのユーザ名を入力して、**[次へ]** をクリックします。 **[完了]** をクリックします。
既存サイトの変更	**[OK]** をクリックして開いているダイアログボックスを閉じます。

注意: Secure Shell サーバと FTP サーバの両方に認証を行う必要があります。

[トンネリング] タブ ([Reflection Secure Shell の設定] ダイアログボックス)

表示方法『15ページ 』

ポートの転送『146ページ 』によって、SSH トンネルを通じて TCP/IP トラフィックを転送できます。これにより、Reflection Secure Shell クライアントを使用して、それ以外の場合には安全でない TCP/IP チャネルを経由して送信されているデータを保護できます。

次のオプションがあります。

[X11 トンネル接続] リモート X11 ポートから送信されるデータはすべて、安全なトンネルを介して正しいローカルポートに自動的に転送されるように指定します。

[ゲートウェイポートを許可] ゲートウェイポートを有効にします。リモートホストをローカル転送ポートに接続できます。 既定で、Reflection Secure Shell は、ローカルポート転送をループバックアドレスに結合します (これは、「ローカルホスト」を使用することに相当します)。これによって、ほかのリモートホストが、転送ポートに接続できないようにしています。**[ゲートウェイポートを許可]** は、Reflection Secure Shell がローカルポート転送をローカルのイーサネットアドレス (IP アドレス、URL、DNS 名など) に結合して、リモートホストが転送ポートへ接続できるようにすることを指定するために使用できます。

この設定を有効にする場合は注意が必要です。この設定によって、リモートホストで、認証なしにシステムで転送されたポートを使用できるようになるため、ネットワークと接続の安全性が低下します。

[ポートのローカル転送] 構成したローカルポートの転送を表示します。**[ローカルポート転送]**『66ページ 』ダイアログボックスを開くには **[追加]** をクリックします。

[ポートのリモート転送] 構成したリモートポートの転送を表示します。**[リモートポート転送]**『67ページ 』ダイアログボックスを開くには **[追加]** をクリックします。

注意

- このダイアログボックスで構成した設定は、*Secure Shell 構成ファイル*『19ページ 』に保存されます。また、任意のテキストエディタを使用して、このファイルを編集して Secure Shell 設定を構成することもできます。
- この構成ファイルの内の設定は、現在指定されている *SSH 構成セクション*『116ページ 』用に保存されます。

[ローカルポート転送] ダイアログボックス

表示方法

1 **[Reflection Secure Shell の設定]** ダイアログボックスを開きます。

2 **[トンネリング]** タブをクリックします。

3 **[ポートのローカル転送]** の **[追加]** をクリックします。

ローカルポート転送を構成するには、このダイアログボックスを使用します。 指定したローカルポートへの送信データは、安全なトンネルを介して、指定したリモートホストのポートへ転送されます。

以下のオプションをすべて指定する必要があります。

[ローカルポートの転送] PC の使用可能なポートを指定します。 このポートに送信されたデータは、SSH トンネルを介して転送されます。

[転送先のホスト] の [名前] データの送信先ホストコンピュータを指定します（localhost を指定して、すでに Secure Shell 接続を確立した同じリモートホストの異なるポートにデータを転送できます）。

[ポート] データの送信先のリモートホストのポートを指定します （**[リモートデスクトップにトンネル接続する]** を選択した場合は、Reflection for Secure IT で正しいリモートポートが自動的に構成され、このボックスは選択できなくなります）。

[転送の種類] **[TCP]** と **[FTP]** の 2 つのオプションがあります。 **FTP** クライアントとサーバ間で通信を転送する場合を除いて、**TCP** を使用してください。

構成可能なオプション

[リモートデスクトップにトンネル接続する] Windows リモートデスクトップセッションにトンネル接続する場合は、このチェックボックスをオンにします。 このオプションをオンにするとその他のオプションが選択できなくなり、Reflection for Secure IT で自動的にセッション転送設定が正しく構成されます。

[Reflection FTP を使用] このボタンは、**[転送の種類]** が [FTP] に設定されている場合のみ表示されます。 クリックすると、**[起動するアプリケーション]** に、Reflection FTP クライアントを起動して FTP 通信をトンネル接続するための正しい値が自動的に入力されます。

[起動するアプリケーション] の [名前]	Secure Shell 接続確立後に、Reflection for Secure IT が自動的に起動するアプリケーション（メールクライアント、FTP クライアント、または Web ブラウザ）の名前を入力します。 安全なトンネルを使用するには、**[ローカルポートの転送]** に設定したポートに接続するようにアプリケーションを構成する必要があります。 一部のアプリケーションでは、コマンドライン引数を使用してこのように構成できます。 引数は **[引数]** テキストボックスに指定します。
[引数]	指定したアプリケーションが起動した時に使用するオプションのコマンドライン引数を指定します。

注意： ポート転送の設定は、現在指定されている *SSH 構成セクション*『116ページ 』に保存されます。

[リモートポート転送] ダイアログボックス

表示方法

1 **[Reflection Secure Shell の設定]** ダイアログボックスを開きます。

2 **[トンネリング]** タブをクリックします。

3 **[ポートのリモート転送]** の **[追加]** をクリックします。

リモートポート転送を構成するには、このダイアログボックスを使用します。 指定したリモートポートからの送信データは、安全なトンネルを介して、指定したローカルコンピュータのポートへ転送されます。

以下のオプションをすべて指定する必要があります。

[リモートサーバポートの転送]	ホストコンピュータのポートを指定します。 このポートから送信されたデータは、SSH トンネルを介して PC に転送されます。
[名前]	データの送信先のローカルコンピュータを指定します。
[ポート]	データの送信先のローカルホストのポートを指定します。

注意： ポート転送の設定は、現在指定されている *SSH 構成セクション*『116ページ 』に保存されます。

マルチホップ Secure Shell セッションの構成

一連の Secure Shell サーバによって安全な接続を確立する必要がある場合、マルチホップ接続を使用します。 これは、直接リモートサーバにアクセスすることはできないが、中間サーバを介してアクセスすることができるネットワーク構成で役に立ちます。 ここでは、そのような一連のサーバについて示します。 Windows ワークステーションにはサーバ C への安全なアクセスが必要ですが、サーバ B にもサーバ C にも直接接続することはできません。 サーバ A はサーバ B に接続でき、サーバ B はサーバ C に接続できます。

Windows ワークステーション ➡ サーバ A ➡ サーバ B ➡ サーバ C

マルチホップ一覧を構成すると、Reflection for Secure IT では一連の安全なトンネルを確立することにより、安全なエンドツーエンド接続を作成します。 各トンネルは既存のトンネル内に確立され、チェーンに沿ってさらにトンネルが確立されます。

チェーン内の最後のサーバは、最初の Secure Shell 接続の設定時に指定したホストになります。 その他のサーバを順番に（クライアント側を基準に上から下に）マルチホップサーバー一覧に追加します。 以下の手順では、この方法について説明します。

マルチホップセッションを構成するには

1 最終宛先ホスト（この例では ServerC）に対し、Reflection Secure Shell セッションを構成します。

2 **[Reflection Secure Shell の設定]**『15ページ 』ダイアログボックスを開きます。

3 **[マルチホップ]** タブをクリックします。

4 [**追加**] をクリックして、構成内の最初のマルチホップサーバ（この例では ServerA）に対し接続を構成します。

   a) [**ホスト名**] には、このホップの宛先ホスト（この例では ServerA）を指定します。

   b) （オプション）元のホスト接続用に指定したユーザとは異なるユーザ名がホストで必要な場合、[**ユーザ名**] の値を指定します（この例では、ServerA と ServerC で異なるユーザ名が必要な場合に、ユーザ名を指定する必要があります）。

   c) （オプション）ホストで Secure Shell 接続用にポート 22 が使用されない場合は、[**ポート**] の値を変更してください。

   d) （オプション）[構成] をクリックするか、[*SSH 構成セクション*『116ページ 』] を指定して、この接続で既定以外の Secure Shell 設定を使用します。

   e) **[OK]** をクリックします。

5 [**追加**] を再びクリックして、追加のマルチホップサーバ（この例では ServerB）に対する接続を構成します。

注意： この接続を別のアプリケーション（ブラウザまたはメールクライアントなど）のデータのトンネリングに使用している場合、**[トンネリング]**『65ページ 』タブをクリックしてポート転送を構成します。例えば、メールサーバがサーバ C で実行されている場合、このマルチホップの構成後、以下のように新しいローカルポート転送を作成できます。**[ローカルポートの転送]** に、未使用のポート（例 1110）を指定し、リモートホストの **[名前]** に **localhost**（このコンテキストで「localhost」は、上記の例の一連のサーバ C の最後のサーバを示します）を入力し、メールサーバポートと同じ **[ポート]** 値（通常 110）を設定します。 Reflection マルチホップトンネルが確立されると、localhost:1110 に接続するようにローカルのメールクライアントを構成することによって、メールサーバに安全にアクセスできるようになります。

[マルチホップ] タブ ([Reflection Secure Shell の設定] ダイアログボックス)

表示方法『15ページ 』

このタブを使って、*マルチホップ*『67ページ 』Secure Shell セッションを構成します。

次のオプションがあります。

[マルチホップサーバ] マルチホップシーケンス内のサーバを表示します。Reflection は、指定されたローカルポートからリモートサーバ上の指定されたポートへの新しい SSH トンネルを確立します。一覧の各接続は、その上の接続で確立されたトンネルを介して送信されます。矢印ボタンを使用して一覧の順序を変更できます。

[追加] **[マルチホップサーバの構成]**『69ページ 』ダイアログボックスを使用して一覧に新しいサーバを追加します。

[変更] 選択したサーバを変更します。

[削除] 選択したサーバを削除します。

注意

- このダイアログボックスで構成した設定は、*Secure Shell 構成ファイル*『19ページ 』に保存されます。また、任意のテキストエディタを使用して、このファイルを編集して Secure Shell 設定を構成することもできます。
- この構成ファイルの内の設定は、現在指定されている *SSH 構成セクション*『116ページ 』用に保存されます。

[マルチホップサーバの構成] ダイアログボックス

表示方法

1 **[Reflection Secure Shell の設定]** ダイアログボックスを開きます。

2 **[マルチホップ]** タブをクリックします。

3 **[追加]** をクリックします。

このダイアログボックスで、*マルチホップ*『67ページ 』の一覧にサーバを追加します。 次のオプションがあります。

[ローカルポートの起動] これは、ローカルの Windows ワークステーションのポートです。 マルチホップ接続は、指定のポート (使用可能な場合) から転送されます。 ポートが使用中の場合は、使用可能なポートが見つかるまでポート番号が増分されます。

[ホスト名] 送信データを通過させるホストコンピュータを指定します。

[構成] [Reflection Secure Shell の設定] ダイアログボックスを開きます。このダイアログボックスでは、このトンネルに既定以外の設定を構成できます。 注意: 既定以外の設定を構成する別の方法として、この接続に **[SSH 構成セクション]** を指定するやり方もあります。

[ポート] データの送信先リモートホストのポートを指定します。 既定では、多くの SSH サーバで使用されているポートである「22」が設定されています。

[ユーザ名] (オプション) 元の接続用に指定したユーザとは異なるユーザ名がホストで必要な場合、ここで名前を指定します。

[SSH 構成セクション] (オプション) この接続に使用する *SSH 構成セクション* 『116ページ 』を指定します (Reflection では、既定でホスト名が使用されます)。

１０　章

ホスト変数とコマンド

この章の内容

[ホストデータ] タブ ([Reflection Secure Shell の設定] ダイアログボックス)

表示方法『15ページ 』

[ホストデータ] タブを使用して、サーバ上で環境変数を設定したり、コマンドを実行したりできます。

次のオプションがあります。

環境変数

[追加]　**[新規環境変数]** ダイアログボックスを開きます。ここで、新しい変数と値を指定できます。

[変更]　選択した変数を編集します。

[削除]　選択した変数を削除します。

リモートコマンド

[コマンド]　リモートサーバで実行する 1 つまたは複数のコマンドを指定します。UNIX サーバへの接続時には、セミコロン (;) を使用して複数のコマンドを区切ります。Windows サーバへの接続時には、アンパサンド記号 (&) を使用してコマンドを区切ります。接続の確立後、サーバは指定したコマンドを実行 (または実行を試行) し、セッションが終了します。サーバは、クライアントから受信したコマンドの実行を許可するよう構成されている必要があります。

コマンドは、正しい形式でサーバに指定する必要があります。例えば、UNIX サーバのディレクトリー覧を取得するには、次のように指定します。

```
ls > list.txt
```

Windows サーバでは、同等のコマンドは、Windows サーバの構成方法に応じて、次のどちらかになります。

```
dir > list.txt
cmd /c dir > list.txt
```

11 章

プロキシサーバ

この章の内容

[プロキシ] タブ ([Reflection Secure Shell の設定] ダイアログボックス)

表示方法『15ページ 』

Reflection Secure Shell セッションでのプロキシの使用を有効にするには、[プロキシ] タブを使用します。

次のオプションがあります。

[なし]	プロキシは設定されていません。(既定の設定)。
SOCKS	**[SOCKS]** を選択すると、SOCKS プロキシを介した Secure Shell 接続を構成できます。
HTTP	**[HTTP]** を選択すると、HTTP プロキシを介した Secure Shell 接続を構成できます。
[構成]	プロキシサーバの設定を構成します。

注意:

- Secure Shell 接続では、プロキシの使用は、Secure Shell 構成ファイルの [プロキシ] 設定を使用し、現在指定されている *SSH 構成セクション*『116ページ 』に対して有効になります。プロキシのサーバアドレスは、ユーザごとに Windows レジストリに保存され、すべての Reflection セッションに適用されます。
- Reflection FTP クライアントの **セキュリティのプロパティ]** ダイアログボックスには、SOCKS の構成と Secure Shell の構成の両方のタブが含まれています。**[Secure Shell]** タブの **[Reflection Secure Shell を使用する]** を有効にした場合、[セキュリティのプロパティ] ダイアログボックスの [SOCKS] タブを使用して、SOCKS プロキシ接続を構成することはできません。SOCKS プロキシを構成するには、[Reflection Secure Shell 設定] ダイアログボックスの [プロキシ] タブの [SOCKS] 設定を使用します。

---

12 章

トラブルシュート

この章の内容

Secure Shell 接続のトラブルシュート

Secure Shell 接続に問題がある場合、問題の原因は Reflection がホストを検索できない、またはホスト認証またはユーザ認証のいずれかの問題の可能性があります。

ログファイルの使用

接続の問題が*ホスト認証*『25ページ 』にある場合、Reflection クライアントの*ログファイル*『76ページ 』に有効な情報を見つけることができます。

問題が*ユーザ認証*『27ページ 』にある場合、Secure Shell サーバの管理者への問い合わせが必要な場合があります。 ユーザ認証の問題は共通しており、失敗したユーザ認証に関する詳細な情報は、クライアントログではなくサーバデバッグログでのみ使用できます。 Secure Shell プロトコルでは、失敗した認証試行に関する固有の情報をクライアントに提供しないようになっています。 これは攻撃者がエラーメッセージを使用できないようにし、認証失敗の理由を判断して攻撃の成功する可能性を減らすためです。

トラブルシュートの候補

パスワードの認証

- パスワードが誤っています。 Caps Lock が無効であることを確認します。
- パスワードの期限が切れています。 パスワード認証ではなくキーボード対話型認証を使用して、パスワード更新を有効にしなければならない可能性があります。
- パスワードメッセージが表示されない場合、パスワード認証が無効の可能性があります。

公開鍵認証

- ユーザの公開鍵が、ホストの正しい位置にアップロードされていません。
- ユーザの公開鍵が正しい位置にアップロードされていますが、所有権またはファイル許可が間違っています。
- 鍵がパスフレーズで保護され、誤ったパスフレーズを入力しています。
- [Reflection Secure Shell の設定] ダイアログボックスの **[ユーザ鍵]**『35ページ 』タブで認証に選択した鍵が誤っています。
- 特に古いバージョンの OpenSSH を実行するサーバへの接続を試行している場合、選択した公開鍵が多すぎます。

証明書認証

- ホストの認証に使用される証明書が使用できません。 Reflection の信頼されたルート格納場所と Microsoft の信頼されたルート格納場所と中間格納場所を確認します (Microsoft の格納場所の使用が無効になっている場合、証明書を Reflection 格納場所にする必要があります)。
- ユーザの認証に使用される証明書が使用できません。 Reflection の個人の格納場所と Microsoft の個人の格納場所を確認します。
- ホストまたはユーザの認証に使用される証明書の有効期限が切れています。
- *[証明書のホスト名と対象ホスト名が一致するかどうかを確認する]*『48ページ 』がオンになっていますが、この接続に指定したホスト名が証明書のホスト名と一致していません。
- *証明書取り消しの確認が有効*『48ページ 』で、Certificate Revocation List が使用できません。
- *証明書の取り消しの確認が有効*『48ページ 』で、ホスト証明書が失効しています。

鍵交換

- 鍵交換中に「fatal: dh_gen_key: group too small: 1024 (2*need 1024)」エラーが表示されます。 鍵交換アルゴリズムを変更して、diffie-hellman-group14-sha1 を他のアルゴリズムよりも前に配置します。

Secure Shell ログファイルの使用

ログファイルには、Secure Shell 接続をトラブルシュートするために使用できる情報が含まれています。

注意:[記録内容]設定を使用すると、Secure Shell ログに記録する情報の量を指定することができます。この設定は、*[Reflection Secure Shell の設定] ダイアログボックスの [全般] タブ*『15ページ 』から使用できます。

Reflection for Secure IT クライアントからログファイルを使用するには

1 トレースをオンにします (**[接続]** - **[トレース]** - **[トレースの開始]**)。

2 接続を行います。

3 トレースをオフにします (**[接続]** - **[トレース]** - **[トレースの終了]**)。

4 トレースを処理します (**[接続]** - **[トレース]** - **[トレースの処理]**)。

5 **[ネットワークプロトコルの詳細]** を選択して **[OK]** をクリックします。

6 **[Logs]** フォルダでトレースファイル (*.rev) を選択し、**[開く]** をクリックします。

7 ログ出力のファイル名と形式を選択し、**[OK]** をクリックします。

FTP クライアントのログファイルを使用するには

- 次のいずれかを実行します。

目的	選択
ファイルへのログ情報の送信	**[ツール]** > **[記録の開始]** で、**[ファイルの種類]** を「診断ファイル (*.txt)」に変更します。
FTP コマンドウィンドウのログ情報の表示	**[表示]** > **[コマンドウィンドウ]**。

Reflection for Secure IT の問題解決のヘルプ

問題:Windows 7、Windows Server 2008、または Windows Vista で Reflection for Secure IT を実行した時、ヘルプが表示されない。

Reflection for Secure IT アプリケーションヘルプ (*.hlp) は、Windows ヘルププログラム (WinHlp32.exe) を使用しています。新しい Windows オペレーティングシステムには、このプログラムのサポートが含まれていません。

Reflection for Secure IT のマニュアルをオンラインで表示するには、**[ヘルプ]** - **[ウェブ上のヘルプトピック]** をクリックします。

ヘルプファイルを新しい Windows システムで有効にするには、Microsoft Download Center から WinHlp32.exe をダウンロードし、インストールします。

ダウンロードの場所と Reflection ヘルプに対応するようにシステムを構成する方法についての詳細は、*技術ノート 2294*
『*http://support.attachmate.com/techdocs/2294.html*』を参照してください。

１３　章

インストールのカスタマイズと配布

この章の内容

管理者用インストール

インストール

以下の 2 種類のインストールを実行できます。

実行するインストール	目的
ワークステーションインストール	少数のコンピュータに Reflection for Secure IT をインストールする
管理者用インストール	配布元として使用できる管理者用インストールポイントを作成する

これらのインストールは、インストールウィザードのグラフィカルインタフェースまたはコマンドラインから実行できます。 また、MSI を使用して Reflection for Secure IT を直接インストールすることもできます。

インストールと配布の計画

Reflection for Secure IT のインストールと配布にはさまざまな方法があります。どの方法を選ぶかは、通常、承認されている業務プロセス、配布の規模、配布に使用するツール、およびカスタムインストールをするかどうかなどのさまざまな要因で決まります。

例えば小規模の配布では、Attachmate インストールプログラムを使用して数台のワークステーションに Reflection for Secure IT をインストールするだけかもしれませんが、企業規模の配布を行うには、膨大なカスタマイズとテストが必要です。

要件に応じて、以下のいずれかの方法を使用してください。

- **各ワークステーション上でのワークステーションインストールの実行**
  Reflection for Secure IT のファイルをすべて PC のハードディスクにインストールします。この方法は、Reflection for Secure IT をインストールするコンピュータの数が少なく、インストールをカスタマイズする必要がない場合に適しています。
- **基本配布の実行**
  管理者用インストールを行って、Reflection for Secure IT のファイルを管理者用インストールポイントにコピーします。この作業を、管理者用インストールイメージを作る、と呼ぶこともあります。次に、配布ツールを使用してこのファイルにアクセスしてワークステーションに配布するパッケージを作成します。基本配布は、インストールのカスタマイズを必要とせず、Reflection for Secure IT を配布するワークステーションの数が多い時に適した方法です。

- **カスタム配布の実行**
  管理者用インストールを行って、Reflection for Secure IT のファイルを管理者用インストールポイントにコピーします（基本配布の場合と同じです）。次に、インストールの方法、外観、およびエンドユーザのコンピュータの動作をカスタマイズします。カスタム配布では、配布するワークステーションの台数に制限はありません。 この場合は、ユーザに、カスタマイズしたファイルを提供することができます（例えば、製品によってはこれらのファイルにワークスペースやセッションドキュメントが含まれます）。

管理者用インストールポイントの作成

配布の環境を準備するには、管理者用インストールポイントを作成する必要があります。管理者用インストールポイントを作成するには、ネットワーク共有（通常はファイルサーバ）上にある Reflection for Secure IT の管理者用インストールイメージをインストールします。管理者用インストールイメージは、DVD に含まれるイメージと同様の、アプリケーションのソースイメージです。管理者用インストールイメージには、カスタム設定に使用する管理ツールと、Reflection for Secure IT のインストールに必要なすべてのファイルが含まれます。

警告：この操作手順では、**[詳細設定]** タブと **[ファイルの場所]** タブのみを使用してください。ほかのタブで作成された構成は無視されます。

注意：

- ワークステーションに Reflection for Secure IT をインストールする前に、管理者用インストールポイントを作成することをお勧めします。管理者用インストールポイントを作成すると、ワークステーションをインストールする際にこれを使用できるようになります。
- Attachmate インストールプログラムのグラフィカルインタフェースでなくコマンドラインを使用する場合は、以下の方法でコマンドラインから管理者用インストールイメージを作成できます。

  ```
  path_to_setup_file\Setup.exe /install /admin
  TARGETDIR=UNC_path_to_administrative_installation_point
  ```

- 最初にワークステーションに Reflection for Secure IT をインストールする場合は、以下の方法でコマンドラインから管理者用インストールイメージを作成する必要があります。

  ```
  path_to_setup_file_on_your_workstation\Setup.exe /install /admin
  TARGETDIR=UNC_path_to_administrative_installation_point
  ```

管理者用インストールポイントを作成するには

1 ネットワークファイルサーバ上にネットワーク共有を作成します。

2 ダウンロードリンクをクリックして、ダウンロードプログラムを実行します。インストーラファイルの位置を選択して、[次へ] をクリックします。

  指定した位置にファイルが抽出され、Attachmate インストールプログラムが開始します。(ファイルをダウンロード済みの場合は、setup.exe ファイルをクリックしてインストールプログラムを開始します。)

3 **[続行]** をクリックしてライセンスに同意します。

4 **[詳細設定]** タブで、**[管理者用インストールイメージをサーバに作成する]** をクリックします。

5 **[続行]** をクリックして、管理者用インストールイメージに使用するネットワーク共有へ移動します。

重要!ネットワーク共有のパスは、必ず UNC パスとして指定してください (\\share_name\administrative_install_point など)。

6 **[インストール]** をクリックします。

注意: 管理者用インストールイメージは、通常ファイルサーバ上のフォルダに作成されます。ただし、管理者用インストールイメージは、ローカルハードディスクのどのフォルダにでも作成できるので、テストをするのに便利です。

コマンドラインからのインストール

Attachmate インストールプログラムのコマンドラインを使用して、配布イメージや管理者用インストールイメージから Reflection for Secure IT をインストールできます。また、バッチファイルにコマンドラインオプションを含めてインストールパラメータを事前設定し、Reflection for Secure IT のインストール中のユーザの対話を制限することもできます。無人インストールを行うため、インストールのダイアログボックスが表示されないようにすることもできます。

また、コマンドラインオプションを使用して、Reflection for Secure IT をユーザごとにインストールするよう準備することもできます。通常、MSI のコマンドラインオプションはすべて Attachmate インストールプログラムのコマンドラインから使用できます。

コマンドラインからインストールするには

- コマンドプロンプト、または [スタート] - [ファイル名を指定して実行] コマンドで setup.exe ファイルのあるディレクトリに移動して、以下のいずれかを実行します。
  - 管理者用インストールイメージを作成するには、以下を入力します。

    `setup.exe /install /admin TARGETDIR=`*`path`*

    path の部分には、サーバ上の管理者用インストールイメージへのパスが入ります。

    または
  - 一般的な設定でワークステーションにインストールするには、以下のように入力します。

    `setup.exe /install INSTALLDIR=`*`path`*

    path には、インストールディレクトリのパスが入ります (INSTALLDIR=*path* はオプション)。

注意:インストールをカスタマイズするためのコマンドラインオプションの一覧を表示するには、setup.exe ファイルのあるディレクトリに移動し、以下のように入力します。

```
setup.exe /?
```

MSI を使用して直接インストールするには

コマンドプロンプトまたは、[スタート] - [ファイル名を指定して実行] コマンドで、msi ファイルのあるディレクトリに移動して、以下のように入力します。

`msiexec.exe /i `*`installation_file_name`*`.msi`

例えば、Reflection for Secure IT Windows クライアント バージョン 7.2 をインストールするには、次のコマンドを使用します。

```
msiexec.exe /i rsshc720.msi
```

インストールのログ記録

既定では、インストーラによってログファイルが作成され、インストールが正常に完了すると、ファイルは削除されます。 インストールログファイルは、インストールに関する詳細情報を提供するもので、ユーザの一時ディレクトリ（%tmp%）に保存され、atm で始まるファイル名が自動生成されます。このフォルダを開くには、[スタート] - [ファイル名を指定して実行] コマンドで、%tmp% を入力します。

インストールログファイルを作成または無効にするには

1 Attachmate インストールプログラムを実行します。

インストール元	手順
ダウンロードサイト	ダウンロードリンクをクリックして、ダウンロードしたプログラムを実行します。インストーラファイルの場所を選択して **[次へ]** をクリックします。これにより、ファイルが指定の場所に解凍され、Attachmate インストールプログラムが起動します。
管理者用インストールイメージ	管理者用インストールポイントから、setup.exe ファイルをダブルクリックします。

2 **[詳細設定]** タブで、**[インストールのログファイルを作成する]** をオンまたはオフにします。ログファイルを作成する場合は、**[インストールが正常に完了した場合は、ログファイルを削除する]** を選択すると、インストールが失敗した場合のみログファイルを保持することができます。

3 **[インストール]** をクリックします。

インストールのカスタマイズ

Reflection カスタム設定ツールを使用し、Reflection for Secure IT インストールをカスタマイズできます。下記の手順は、このツールを起動し、使用して、インストールをカスタマイズする方法を示しています。追加情報は、Reflection カスタム設定ツールの **[ヘルプ]** メニューから使用できます。

Reflection カスタム設定ツールを開く

Reflection カスタム設定ツール を実行するためには、最初に管理者用インストールイメージを作成しておく必要があります。

Reflection カスタム設定ツール を開くには

次のいずれかを実行します。

- コマンドラインで、管理者用インストールポイントに移動し、setup /admin と入力します。

  または

- *ACT へのショートカットを設定* 『83ページ 』している場合は、ショートカットをダブルクリックします。

[カスタム設定の選択] ダイアログボックスで、どの *モード* 『83ページ 』を開きたいかを選択するように要求されます。

注意:Reflection カスタム設定ツールを実行している最中に、Attachmate インストールプログラムを実行することはできません。setup.exe プログラムのインスタンスは一度に 1 つしか実行できません。

ACT へのショートカットの設定

既定では、Attachmate カスタム設定ツール (ACT) はコマンドラインからのみ開くことが可能ですが、デスクトップショートカットを作成し、ショートカットプロパティを設定して、このツールを開くことができます。このショートカットの作成はオプションですが、ACT を操作する時間が短縮されるため便利です。

ACT を開くデスクトップショートカットを設定するには

1 管理者用インストールポイントで、setup.exe ファイルを右クリックし、[ショートカットの作成] を選択します。

2 ショートカットを右クリックし、[プロパティ] を選択します。

3 [ターゲット] フィールドで、コマンドラインの最後に /admin オプションを追加します。例えば、次のように入力します。

```
¥¥myServer¥adminInstallPoint¥setup.exe /admin
```

警告:[ターゲット] フィールドのパスが UNC (汎用命名規約) 形式で参照されていることを確認してください。パス名にはドライブ文字を使用しないでください。ドライブ文字を使用すると、ほかのワークステーションでショートカットを使用する際に問題が発生する場合があります。

4 ショートカットの名前を変更してから、ワークステーションのデスクトップおよび管理者用インストールポイント用に使用しているサーバに保存します。

カスタム設定の種類の選択

Reflection カスタム設定ツールを開いたら、トランスフォームやコンパニオンインストールパッケージを作成したり、既存のファイルを開いたりすることができます。

カスタム設定の種類を選択するには

1 **[カスタム設定の選択]** ダイアログボックスで、カスタム設定の種類を選択します。

目的	選択する項目
新規トランスフォーム (.mst) を作成する	**[以下の製品のセットアップカスタム設定ファイルを新規作成する]** (既定)
新規コンパニオンインストールパッケージ (.msi) を作成する	**[コンパニオンインストーラを新規作成する]**
既存ファイルを開く	**[既存のセットアップカスタム設定ファイルまたはコンパニオンインストーラを開く]**

2 **[OK]** をクリックします。

コンパニオンインストールパッケージの作成

Reflection for Secure IT とともにインストールされないファイルをインストールするために、コンパニオンインストールパッケージ (「コンパニオンデータベース」とも呼びます) を作成します。

コンパニオンファイルは、Reflection for Secure IT とは別にインストールされるため、本製品をアップグレードしても削除されません。また、製品を再インストールせずに、追加のサポートファイルを配布することができます。例えば、対応している部門がいくつかあり、各部門用に設定ファイルをカスタマイズする必要がある時、部門ごとにコンパニオンインストールパッケージを作成することができます。

コンパニオンインストールパッケージは以下に対応しています。

- ファイルとインストールディレクトリの指定
- 選択したファイルのショートカット
- アプリケーションのカスタム設定の追加
- 主要インストーラとの連結（連結は .mst ファイルに指定する必要があります。）
- 主要インストールに影響せずに削除（主要インストールも、コンパニオンパッケージが追加したファイルに影響を与えずに削除できます。）

コンパニオンインストールパッケージは、MSI パッケージ__に対応している環境においてインストールできます。パッケージには、標準の進行状況バー以外、組み込まれているインタフェースはありません。

コンパニオンインストールパッケージの作成

Reflection for Secure IT とともにインストールされないファイルをインストールするために、コンパニオンインストールパッケージ（「コンパニオンデータベース」とも呼びます）を作成します。

コンパニオンインストールパッケージを作成するには

1 Reflection をインストールしたワークステーションで、*デスクトップショートカット*『83ページ』から Attachmate カスタム設定ツールを開きます。または、次のようにコマンドラインから開きます。

path_to_setup\setup.exe /admin

2 **[カスタム設定の選択]** ダイアログボックスで、**[コンパニオンインストーラを新規作成する]** を選択し（または既存の MSI を開き）、**[OK]** をクリックします。

3 ナビゲーション画面で、**[パッケージ情報の指定]** を選択します。

4 **[名前の追加と削除]** ボックスおよび **[所属名]** ボックスに、Microsoft Windows の [プログラムのアンインストールまたは変更] パネルに表示させるインストールと発行者の名前を入力します。

注意: Windows Vista および Windows 7 の *[プログラムのアンインストールまたは変更]* リストは、以前のバージョンの Windows バージョンの *[プログラムの追加と削除]* リストに類似しています。 アクセスするには、[コントロールパネル] から [プログラムと機能] を選択します。

---

5 ナビゲーション画面で、**[インストール場所の指定]** を選択します。

6 **[インストールの種類]** で、ファイルをインストールする対象（全ユーザまたは 1 ユーザのみ）を選択します。

選択項目	コンパニオンインストーラパッケージを設定するには
[マシンの全ユーザにインストール]	コンピュータにログオンしている全ユーザがファイルを使用できます。 ファイル、マクロ、その他の構成ファイルを全ユーザが使用できるように設定する場合は、このオプションを使

	用します。
[インストールするユーザのみにインストール]	インストールユーザのみがファイルを使用できます。 重要!ユーザの個人用データフォルダにインストールする必要があるファイルを配布する場合は、このオプションを選択してください。

7 **[既定のインストールフォルダ]** リストで、ファイルをインストールするフォルダを選択します(ファイルを追加する時に別のフォルダを指定しない場合、ファイルはこのフォルダに配布されます)。

注意: このリストで使用できるフォルダは、選択したインストールの種類によって異なります。**[インストールの種類]** フォルダオプションでは、(インストール後の) ファイルへのアクセスを指定します。 このオプションは、インストールするフォルダのみに適用されます。ファイルを追加した後は、オプションを変更することはできません。

8 **[既定のショートカットフォルダ] リストで、プログラムのショートカットをインストールするフォルダを選択します (ファイルを追加する時に別のフォルダを指定しない場合、ショートカットはこのフォルダに配布されます)。**

注意:フォルダを参照するリスト項目 (`[ProgramMenuFolder]` など) は定義済みのフォルダキーワードです。カスタム設定ディレクトリを作成するには、一般的なディレクトリ構文 (`[ProgramFilesFolder]¥マイ フォルダ`など) に従って新規フォルダ名を追加します。または、ターゲットマシンに場所が存在していることがわかっている場合にはフルパス (`C:¥Program Files¥マイ フォルダ`など) を入力することもできます。

注意: **インストールの種類**のオプションは、(インストール後の) ファイルへのアクセスを指定します。このオプションは、インストールするフォルダのみに適用されます。ファイルを追加した後は、オプションを変更することはできません。

9 ファイルまたはショートカットを、追加、更新、または削除するには、ナビゲーション画面で **[ファイルの追加]** を選択します。

コンパニオンインストールの設定プロパティやユーザ設定を変更することもできます。

コンパニオンパッケージの作成が完了したら、パッケージを単独で、または Reflection for Secure IT とともに配布することができます。パッケージは、Microsoft の [プログラムの追加と削除] ユーティリティで削除するまで、ユーザのコンピュータ内に残ります。

コンパニオンインストーラによるカスタム設定のインストール

Reflection カスタム設定ツール を使用し、カスタマイズされた Reflection for Secure IT を構成できます。これを行うには、カスタム設定ファイルをインストールするために 1 つまたは複数のコンパニオンインストーラパッケージを作成し、インストールにコンパニオンパッケージを追加することができます。ファイルをインストールするためのユーザ固有の場所とグローバルな場所の両方を指定できます。

注意:Reflection for Secure IT が使用する構成ファイルの名前と場所については、手順の後にある一覧を参照してください。

Reflection for Secure IT 設定ファイルをインストールするためのコンパニオンパッケージを作成するには

1 *管理者用インストールイメージを作成*『80ページ 』します。

2 Reflection をインストールしたワークステーションで、*デスクトップショートカット*『83ページ 』から Attachmate カスタム設定ツールを開きます。または、次のようにコマンドラインから開きます。

```
path_to_setup\setup.exe /admin
```

3 **[カスタム設定の選択]** ダイアログボックスで、**[コンパニオンインストーラを新規作成する]** を選択し（または既存の MSI を開き）、**[OK]** をクリックします。

4 ナビゲーション画面で、**[パッケージ情報の指定]** をクリックします。このタブを使用して、パッケージが Windows の [プログラムの追加と削除] 一覧で使用するプログラム名を指定します。また、所属名も指定します。

5 ナビゲーション画面で、**[インストール場所の指定]** をクリックします。このパネルを使用して、すべてのユーザ用にファイルをインストールする（既定）か、作成しているパッケージをインストールするユーザのためだけにファイルをインストールするかを指定します。また、このパネルで、既定のインストール場所を指定できます。

6 ナビゲーション画面で、**[ファイルの追加]** をクリックします。

7 **[ファイルの追加先]** で、インストール先の場所を指定します。Reflection for Secure IT が使用するファイルの一覧とファイルの場所については、手順の後にある一覧を参照してください。

8 （オプション）**[ショートカットを含める]** をクリックして、インストールされたファイルを開くためにユーザが使用できるショートカットをインストールします。例えば、設定ファイル（*.r3w）をインストールしている場合、そのファイル内の設定を使用して Reflection を起動するためのショートカットをインストールすることができます。**[ショートカットの構成]** パネルを使用すると、ショートカットをインストールする場所を指定することができます。

9 **[追加]** をクリックし、インストールに追加するファイルを指定し、**[開く]** をクリックします。

10 **[ファイル]** - **[名前を付けて保存]** コマンドをクリックして、インストーラファイルの名前（ReflectionSettings.msi など）を入力します。

ユーザ固有のファイルと場所

ファイル名	ファイルの追加先
*.r3w	[PersonalFolder]\Attachmate\Reflection
	注意：これは、Reflection for Secure IT 設定ファイル用の既定の場所です。ほかの場所にある設定ファイルも使用できます。
config	[PersonalFolder]¥Attachmate¥Reflection¥.ssh
	注意：このファイルについては、「*Secure Shell クライアント構成ファイル*『19ページ 』」を参照してください。
known_hosts	[PersonalFolder]¥Attachmate¥Reflection¥.ssh
	注意：このファイルについては、「*既知のホストファイル*『39ページ 』」を参照してください。
pki_config	[PersonalFolder]\Attachmate\Reflection\.pki
	注意：このファイルは、Reflection 証明書マネージャ設定を構成します。

ファイル名	ファイルの追加先
trust_store.p12	[PersonalFolder]\Attachmate\Reflection\.pki
	注意:このファイルは、Reflection の信頼された認証局を構成します。
Settings.rfw	[PersonalFolder]¥Attachmate¥Reflection
	注意:これは、FTP クライアント設定ファイル用の既定の場所です。また、xml ファイルを使用して FTP クライアントを構成することもできます。xml 設定ファイルを使用する利点は、ユーザ名とパスワードのようなユーザ固有の情報を含めずに設定を配布できることです。

rftp.xml	[PersonalFolder]¥Attachmate¥Reflection
	注意:このファイルは、FTP クライアント **[エクスポート]** コマンドを使用して作成できます。この場所にある設定は、ユーザが初めて FTP クライアントを実行する時に Settings.rfw ファイルに移行されます。
rsckrb5.xml	[AppDataFolder]¥Attachmate¥Reflection
	注意:このファイルは、Reflection Kerberos マネージャ **[設定のエクスポート]** コマンドを使用して作成できます。この場所にある設定は、ユーザが初めて Reflection Kerberos マネージャ を実行する時、または Reflection Kerberos を使用するように構成された任意の Reflection クライアントを実行する時、Windows レジストリに移行されます。

グローバルなファイルと場所

ファイル名	ファイルの追加先
ssh_config	[CommonAppDataFolder]\Attachmate\Reflection
	注意:これは、グローバルな Secure Shell クライアント構成ファイルです。
ssh_known_hosts	[CommonAppDataFolder]\Attachmate\Reflection
	注意:これは、グローバルな既知のホストファイルです。
pki_config	[CommonAppDataFolder]\Attachmate\Reflection\.pki
trust_store.p12	[CommonAppDataFolder]\Attachmate\Reflection\.pki
rftp.xml	[CommonAppDataFolder]¥Attachmate¥Reflection
	注意:この場所にある設定は、各 Windows ユーザが初めて FTP クライアントを実行する時に Settings.rfw ファイルに移行されます。
rsckrb5.xml	[CommonAppDataFolder]¥Attachmate¥Reflection
	注意:この場所にある設定は、各 Windows ユーザが初めて Reflection Kerberos マネージャ を実行する時、または Reflection Kerberos を使用するように構成された任意の Reflection クライアントを実行する時、Windows レジストリに移行されます。

インストールへのコンパニオンインストーラの追加

この手順を使用すると、製品インストールに 1 つまたは複数のカスタムインストーラパッケージを追加することができます。

インストールにコンパニオンパッケージを追加するには

1 Reflection をインストールしたワークステーションで、*デスクトップショートカット*『83ページ』から Attachmate カスタム設定ツールを開きます。または、次のようにコマンドラインから開きます。

path_to_setup\setup.exe /admin

2 **[以下の製品のセットアップカスタム設定ファイルを新規作成する]** を選択し、**[OK]** をクリックします。

3 ナビゲーション画面で、**[インストールの追加とプログラムの実行]** をクリックします。

4 **[追加]** をクリックします。

[プログラムエントリの追加と変更] ダイアログボックスが開きます。

5 **[ターゲット]** 一覧で、コンパニオン .msi ファイルを探して選択します。

6 **[このプログラムをベース製品のインストール後に実行する]** を選択します。

7 **[OK]** をクリックします。

8 **[ファイル]** - **[名前を付けて保存]** をクリックして、トランスフォームを保存します。

注意:トランスフォームを保存すると、カスタム設定ツールは、コンパニオンパッケージをインストールする手順を記載した [RunPrograms] セクションを追加して、Setup.ini ファイルを自動的に更新します。

---

9 Setup.exe を使用してインストールするようにユーザに伝えます。

インストールが完了すると、コンパニオンパッケージが自動的にインストールされます。

FTP クライアント設定のインストール

この手順では、Reflection カスタム設定ツールの **[ユーザ設定の変更]** オプションを使用して、FTP クライアントのカスタム設定をインストールします。開始する前に、管理者用インストールを作成します。このインストールの setup.exe を使用して、Reflection カスタム設定ツールを起動します。また、FTP クライアントがコンピュータにインストールされている必要があります。

設定を構成するには

1 Reflection FTP クライアントを起動します。

2 インストールに含めるサイトおよび設定を構成して、設定を保存します。

コンパニオンインストーラを新規作成するには

1 Reflection をインストールしたワークステーションで、*デスクトップショートカット*『83ページ』から Attachmate カスタム設定ツールを開きます。または、次のようにコマンドラインから開きます。

path_to_setup\setup.exe /admin

2 **[カスタム設定の選択]** ダイアログボックスで、**[コンパニオンインストーラを新規作成する]** を選択し（または既存の MSI を開き）、**[OK]** をクリックします。

3 ナビゲーション画面で、**[パッケージ情報の指定]** をクリックします。このタブを使用して、パッケージが Windows の [プログラムの追加と削除] 一覧で使用するプログラム名を指定します。また、所属名も指定します。

4 ナビゲーション画面で、**[インストール場所の指定]** をクリックします。**[インストールの種類]** で、**[インストールするユーザのみにインストールする]** をオンにします。

5 ナビゲーション画面で、**[ユーザ設定の変更]** をクリックします。

6 Reflection 製品の一覧で Reflection FTP クライアントを選択し、**[定義]** をクリックします。

注意： FTP クライアントがワークステーションにインストールされていないと、**[定義]** ボタンは使用できません。

---

7 配布に含める FTP クライアント設定を選択して、**[OK]** をクリックします。設定がエクスポートされたことを示す確認メッセージが表示されます。**[OK]** をクリックしてこのメッセージを閉じます。

注意： **[ユーザ設定]** をオフにする（既定）と、エクスポートされたファイルには、ユーザ名、パスワードなどのユーザ固有の情報は含まれません。詳細については、ダイアログボックスの **[ヘルプ]** ボタンをクリックしてください。

---

8 **[ファイル]** - **[名前を付けて保存]** をクリックし、インストーラファイルの名前（例えば、FTPClientSettings.msi）を入力します。

カスタム設定したコンパニオンインストーラを配布するには

1 *コンパニオンインストーラを Reflection インストールに追加* 『85ページ 』します。

2 ユーザに Setup.exe を使用してインストールするように指示します。

コンパニオンパッケージによって、ユーザの Application Data フォルダの Attachmate¥Reflection フォルダに XML 設定ファイルがインストールされます。この場所に rftp.xml ファイルがインストールされると、このファイルに構成された設定は、ユーザが初めて FTP クライアントを実行した時に Settings.rfw ファイルに移行されます。

トランスフォームの作成と編集

Reflection カスタム設定ツールを使用すると、主要なインストーラのデータベースを修正する MSI トランスフォームを作成することができます。 また、Reflection カスタム設定ツールを使用してこの製品用に作成した既存の .mst ファイルを開いて編集することもできます。

作成したトランスフォームは、インストールに含める必要があります。トランスフォームは、setup.exe で起動するあらゆるインストールまたはコマンドラインインストール（配布ツールの多くで使用されている）とともに使用することができます。 インストーラがトランスフォームを適用できるのは、インストール実行時だけです。

注意：

- トランスフォームファイルを SETUP.EXE とともに使用するには、**[ユーザインタフェース]** パネルで **[このカスタム設定を、setup.exe を使用した対話型インストールに使用する]** をオンにします。このオプションをオンにしてトランスフォームを保存すると、Reflection カスタム設定ツールによって SETUP.INI ファイルが自動的に更新され、トランスフォームが Reflection for Secure IT のインストールに適用されます。
- 指定したカスタム設定をインストールするには、Reflection for Secure IT のパッケージファイルと同じフォルダにトランスフォームファイルを保存し、インストール中に参照する必要があります。

対応している変更は以下のとおりです。

- インストール場所
- ユーザインタフェースのレベル - サイレントまたは対話型（setup.exe および setup.ini が必要）
- 機能の状態（隠し機能を含む）
- ショートカットの変更または削除
- 旧製品のアップグレード（オプション）
- ほかのインストールとの連結、または主要なインストールの前後にプログラムを実行する（setup.exe および setup.ini が必要）
- Windows インストーラプロパティの変更

トランスフォームの作成

Reflection for Secure IT をカスタマイズして、インストールの方法、外観、およびユーザのコンピュータでの動作を指定します。

インストールのトランスフォームを作成するには

1 Reflection をインストールしたワークステーションで、*デスクトップショートカット*『83ページ』から Attachmate カスタム設定ツールを開きます。または、次のようにコマンドラインから開きます。

path_to_setup\setup.exe /admin

2 **[以下の製品のセットアップカスタム設定ファイルを新規作成する]** を選択し、**[OK]** をクリックします。

3 左側のパネルのリストで項目を選択すると、右側に構成パネルが開くので、そこでカスタマイズを行います。

4 **[ファイル]** - **[名前を付けて保存]** コマンドを選択します。

トランスフォームファイルは .mst ファイルとして保存されますが、その際 Reflection for Secure IT のインストーラパッケージファイルと同じフォルダに保存することを推奨します。

既存のインストールのトランスフォームを変更するには

1 Reflection をインストールしたワークステーションで、*デスクトップショートカット*『83ページ』から Attachmate カスタム設定ツールを開きます。または、次のようにコマンドラインから開きます。

path_to_setup\setup.exe /admin

2 **[既存のセットアップカスタム設定ファイルまたはコンパニオンインストーラを開く]** をオンにし **[OK]** をクリックします。

3 **[ファイルを開く]** ダイアログボックスで、トランスフォームファイルを作成した時に選択した場所を参照し、*transform_name*.mst ファイルを選択します。

4 左側のパネルのリストで項目を選択すると、右側に構成パネルが開くので、そこでカスタマイズを行います。

5 **[ファイル]** - **[名前を付けて保存]** コマンドを選択します。

トランスフォームファイルは .mst ファイルとして保存されますが、その際 Reflection for Secure IT のインストーラパッケージファイルと同じフォルダに保存することを推奨します。

インストールの追加（連結）とプログラムの実行

Reflection for Secure IT では、連結インストールが簡単にできます。主要インストールの前後に、自動的にコンパニオンインストールパッケージを実行するようにインストールを設定することができます。また、ほかのスクリプトやプログラムを実行するように指定することもできます。

注意：このインストールを「変更」する方法は、Setup.exe によって実行されるインストールにのみ適用されます。MSI コマンドラインの方法によるインストールには適用されません。

インストールまたはプログラムを連結するには

1 Reflection をインストールしたワークステーションで、*デスクトップショートカット*『83ページ』から Attachmate カスタム設定ツールを開きます。または、次のようにコマンドラインから開きます。

path_to_setup\setup.exe /admin

2 **[以下の製品のセットアップカスタム設定ファイルを新規作成する]** を選択し、**[OK]** をクリックします。

3 Attachmate カスタム設定ツールのナビゲーション画面で、**[ユーザインタフェース]** を選択し、**[このカスタム設定を、setup.exe を使用した対話型インストールに使用する]** をオンにします。

4 Reflection カスタム設定ツール のナビゲーション画面で、**[インストールの追加とプログラムの実行]** を選択します。

5 **[追加]** をクリックします。

[プログラムエントリの追加と変更] ダイアログボックスが開きます。

6 **[ターゲット]** リストで、プログラムの .exe ファイルまたは .msi ファイルのあるフォルダを入力または選択し、次に実行するファイル名を入力します。

例 msiexec.exe

7 **[引数]** の下に、実行するコマンドライン引数を入力します。

例 /i my_installation.msi

8 プログラムを実行する時期を指定するには、**[このプログラムをベース製品のインストール後に実行する]** または **[このプログラムをベース製品のインストール前に実行する]** のいずれかを選択します。

注意：ほとんどの場合、**[このプログラムをベース製品のインストール後に実行する]** を選びます。**[このプログラムをベース製品のインストール前に実行する]** を選んで、このプログラムが失敗した場合、Reflection for Secure IT はインストールされません。

9 ほかのプログラムや MSI ファイルを追加するには、上記の手順を繰り返します。

10 実行順序を変更するには、**[移動]**（画面の左下部分）の隣にある矢印を使います。プログラムをリストから削除するには、選択して **[削除]** をクリックします。

インストールにトランスフォームを適用する

Reflection for Secure IT のインストール方法をカスタマイズするためにトランスフォームを作成してある場合は、プライマリインストールでトランスフォームを配布する必要があります。トランスフォームは、setup.exe で開始するインストールや、(多くの配布ツールが使用する) コマンドラインからのインストールで使用できます。インストーラがトランスフォームを適用できるのは、インストールの間だけです。

setup.exe で開始するインストールにトランスフォームを追加するには

1 Reflection をインストール済みのワークステーションで、デスクトップのショートカット (ショートカットがページに表示されるよう設定してある場合)、または次のコマンドラインから、Attachmate カスタム設定ツールを開きます。

path_to_setup\setup.exe /admin

2 **[カスタム設定の選択]** ダイアログボックスで **[既存のセットアップカスタム設定ファイルまたはコンパニオンインストーラを開く]** を選択し、**[OK]** をクリックします。

3 **[ファイルを開く]** ダイアログボックスで、トランスフォームファイルの作成時に選択した場所を参照し、トランスフォーム (.mst) ファイルを選択します。

4 **[ユーザインタフェース]** パネルで、**[このカスタム設定を、setup.exe を使用した対話型インストールに使用する]** をオンにします。

このオプションをオンにしてトランスフォームを保存すると、Attachmate カスタム設定ツールが次の行を自動的に SETUP.INI ファイルの [Setup] セクションに追加して SETUP.INI ファイルを更新し、Reflection のインストールにトランスフォームを適用します。

CustomTransform=<your_transform.mst>

5 **[ファイル]** - **[保存]** コマンドをクリックします。([保存] コマンドがグレイ表示されている場合は、[終了] コマンドをクリックするとファイルの保存を求められます。)

これで、トランスフォームが setup.exe ファイル経由でエンドユーザに配布できるようになりました。(ユーザは setup.exe を実行できます。または、setup.exe ファイルをスクリプトから呼び出したり、コマンドラインから開始したりすることができます。)

setup.exe のコマンドラインからのインストールにトランスフォームを追加するには

- コマンドラインからのインストールにトランスフォームを追加することもできます。以下のコマンドライン構文を使用します。

<*path_to_setup*>\setup.exe /install TRANSFORMS=transform.mst

Reflection のグループポリシへの対応

Reflection はグループポリシに対応するため、管理者は追加ツールを使用して、Reflection アプリケーションをカスタマイズして保護できます。

ポリシを使用するには、次に説明するように、最初に Reflection ポリシテンプレートをポリシエディタに追加する必要があります。グループポリシごとに、[プロパティ] シートの [説明] タブに実行内容の説明が表示されます。

Reflection ポリシテンプレートをインストールするには

1 Reflection 配布フォルダ (または CD) の adm フォルダで ReflectionPolicy.adm を検索します。グループポリシを構成するコンピュータからこのファイルを使用できるようになっている必要があります。

注意:テンプレートをグループポリシエディタに追加するには ADM テンプレートファイルが必要ですが、テンプレートの追加後は、ファイルは不要です。

2 グループポリシエディタを実行するには、次のいずれかの方法を使用します。

- コマンドラインで次のように入力します。

  Gpedit.msc

- **[Active Directory のユーザとコンピュータ]** コンソールで **[組織単位]** のプロパティを開き、**[グループポリシ]** タブをクリックして新規ポリシオブジェクトを編集または作成します。

3 **[ユーザの構成]** - **[管理用テンプレート]** - **[テンプレートの追加と削除]** を選択します。

4 [テンプレートの追加と削除] ダイアログボックスで、**[追加]** をクリックし、ファイルを参照して ReflectionPolicy.adm ファイルを検索します。テンプレートを開き、その後、[テンプレートの追加と削除] ダイアログボックスを閉じます。

注意:大半の Reflection ポリシはユーザごとに構成されます。Windows XP では、[ローカルコンピュータポリシ] - [ユーザの構成] - [管理用テンプレート] - [Reflection Settings] (Reflection の設定) でこれらのポリシを表示および編集します。より新しい Windows システムでは、[ローカルコンピュータポリシ] - [ユーザの構成] - [管理用テンプレート] - [従来の管理用テンプレート (ADM)] - [Reflection Settings] (Reflection の設定) でこれらのポリシを表示および編集します。

対応しているポリシ

Reflection for Secure IT および Reflection FTP クライアントでは次のポリシを使用できます。

* Reflection for Secure IT には適用されません。
** Reflection FTP クライアントには適用されません。

Reflection 設定の表示および編集は、**[ローカルコンピュータポリシ]** - **[ユーザの構成]** - **[管理用テンプレート]** - **[従来の管理用テンプレート (ADM)]** - **[Reflection Settings]** (Reflection の設定) で行います。

[Reflection Settings] (Reflection の設定):

[Allow Start Screen] (画面を起動します)
[Allow Reflection to save passwords] (パスワードを保存します) *
[Allow files to be received from host computers] (ホストコンピュータへのファイルの送信を許可します)
[Allow files to be received from host computers] (ホストコンピュータからのファイルの受信を許可します)
[Allow Sessions without Settings Files] (設定ファイルなしでセッションを開始) **
[Language Override] (言語の優先)
[Allow tracing for troubleshooting] (トラブルシューティングの追跡) **
[Migration of settings from F-Secure to Reflection] (設定を F-Secure から Reflection に移行する) **
[Allow unencrypted connections] (非暗号化接続を許可) *
[Folder for the default Auto Update file] (既定の自動更新ファイルフォルダ) **
[Settings only in these Folders] (これらのフォルダの設定のみ) **

[Reflection Settings] (Reflection の設定) - [When Reflection Exits] (Reflection の終了時に):

[Prompt if connected when user exits Reflection] (Refrection の終了時にユーザが接続中である場合は警告する)
[Prompt when exiting all Reflection sessions] (すべての Reflection セッションを終了する場合は警告する)
[If there are unsaved changes] (保存していない変更がある場合) **

[Reflection Settings] (Reflection の設定) - [Application Programming Interfaces] (アプリケーションプログラミングインタフェース)

[Allow scripts and macros on the startup command line] (起動コマンドラインでスクリプトやマクロを許可する) *
[Allow other applications to use Reflection's OLE Automation interface] (Reflection の OLE オートメーションインターフェースで他のアプリケーションを使用する) *
[Allow Reflection FTP Client Scripting] (Reflection FTP クライアントのスクリプトを許可する)*

Secure Shell コマンドラインユーティリティ

Reflection Secure Shell クライアントには、次の DOS コマンドラインユーティリティが含まれています。これらのユーティリティに対応している実行ファイルは、PC の Reflection プログラムファイルと同じ位置にインストールされます。

- *ssh*『96ページ』
- *ssh-keygen*『101ページ』
- *sftp*『104ページ』
- *scp*『108ページ』

以下の追加ユーティリティは、F-Secure から移行中で、F-Secure コマンドラインユーティリティ用に作成されたスクリプトを維持する必要があるお客様向けに提供されています。これらのユーティリティは、F-Secure と同じスイッチセットに対応しています。

注意:F-Secure コマンドラインユーティリティ用に作成されたスクリプトがない場合は、上記のユーティリティを使用することをおすすめします。

- *ssh2*『100ページ』
- *sftp2*『108ページ』
- *scp2*『111ページ』

ssh コマンドラインユーティリティ

構文: ssh [options] [user@]hostname [host command]

ssh コマンドラインユーティリティを使用して、Windows コマンドラインからの Secure Shell 接続が可能です。

注意:

- Reflection では **ssh2** 『100ページ 』ユーティリティも提供されています。**ssh** と **ssh2** の両方を使用して Secure Shell 接続を確立できますが、これらの 2 つのユーティリティで対応している一部のオプションが異なります。**ssh** オプションは、Reflection クライアントのみが対応している一部の追加オプションを含む OpenSSH Secure Shell 実装に基づいています。**ssh2** オプションは、Reflection for Secure IT UNIX クライアントおよび F-Secure クライアントと互換性があります。
- 既存の Secure Shell 接続を再利用できます。ただし、再利用するには、各コマンドラインでこれを明示的に有効にするか、SSHConnectionReUse 環境変数を Yes に設定する必要があります。詳細については、「*Secure Shell セッションにおける接続の再利用* 『28ページ 』」を参照してください。

オプション

-A

認証エージェント転送を有効にします。 これは、ホスト単位で*構成ファイル* 『19ページ 』に指定することもできます。エージェントの転送を有効にする場合は注意が必要です。リモートホストでファイル権限を回避できるユーザは、転送された接続を介してローカルエージェントにアクセスできます。攻撃者は鍵の情報を取得できませんが、エージェントに読み込まれた識別情報を使用して、その鍵で操作を実行して認証を有効にすることができます。

-a

認証エージェントの転送を無効にします。 (既定の設定)。

-b *bind_address*

複数インタフェースまたは別名アドレスを使用して、マシンから送信するインタフェースを指定します。

-c *cipher_spec*

優先順に指定した暗号のカンマ区切りリスト。既定値は、「aes128-ctr,aes128-cbc,aes192-ctr,aes192-cbc,aes256-ctr,aes256-cbc,3des-cbc,blowfish-cbc,cast128-cbc,arcfour128,arcfour256,arcfour」です。接続が *FIPS モード* 『22ページ 』で行われるように設定されている場合、既定値は「aes128-ctr,aes128-cbc,aes192-ctr,aes192-cbc,aes256-ctr,aes256-cbc,3des-cbc」です。

プロトコルバージョン 1 (廃止される可能性があるため、推奨されません) では、単一の暗号の指定が許可されています。対応する値は「3des」、「blowfish」、「des」です。

-C

すべての送信データの圧縮を有効にします。圧縮はモデム回線やほかの低速接続に適していますが、高速のネットワークでは応答速度の低下を招くだけです。

-e *escape_character*

端末セッションのエスケープ文字を設定します。既定の文字はチルダ (~) です。エスケープ文字を「none」に設定すると、使用できるエスケープ文字はなくなり、チルダは他の文字と同様に機能します。 次のエスケープシーケンスを使用できます(波型符号を指定した escape_character で置換します)。

~.接続を終了します。
~R 鍵の変更を要求します (SSH プロトコル 2 のみ)。
~# 転送された接続を一覧表示します。
~?使用可能なエスケープシーケンスを表示します。
~~ エスケープ文字を 2 回入力して、ホストに送信します。

-E *provider*

指定したプロバイダを外部鍵プロバイダとして使用します。

-f

コマンドが実行される直前に、クライアントをバックグラウンドに配置します。

-F *config_file*

この接続に使用する代替構成ファイルを指定します。構成ファイルをコマンドラインで指定した場合、ほかの ***構成ファイル*** 『19ページ 』は無視されます。

-g

ゲートウェイポートを有効にします。リモートホストをローカル転送ポートに接続できます。

-h

コマンドラインオプションに関する簡単な説明を表示します。

-H *scheme*

この接続に使用する ***SSH 構成セクション*** 『116ページ 』を指定します。

-i *key_file*

鍵認証に使用する秘密鍵を指定します。 鍵ファイルは、ホスト単位で***構成ファイル*** 『19ページ 』に指定することもできます。複数の -i オプション (および構成ファイルに指定した複数の鍵) を指定できます。 ファイルまたはパスが空白を含む場合、引用符を使用します。

-k *directory*

config、ホスト鍵、ユーザ鍵ファイルの代替位置を指定します。注意:-k が使用されていると、既知のホストファイルが指定した場所にすでに存在していない場合のみ、その場所からホスト鍵の読み取りと書き込みが行われます。既知のホストファイルが見つからない場合、既定の場所で既知のホストファイルにホスト鍵の読み取りと書き込みが行われます。

-l *login_name*

リモートコンピュータでのログインに使用する名前を指定します。 これを***構成ファイル*** 『19ページ 』に指定することもできます。

-L *localport:remotehost:hostport*

指定されたローカルポートからのデータを、安全なトンネルを介して、指定された宛先ホストおよびポートにリダイレクトします。 詳細については、「ローカルポート転送」を参照してください。 ポート転送を構成ファイルに指定することもできます。管理者としてログインしないかぎり、権限ポート (ポート番号 1024 以下) を転送できません。IPv6 アドレスは、別の構文 port/host/hostport を使用して指定できます。

-m *mac_spec*

この接続に使用する 1 つまたは複数のカンマ区切り MAC (メッセージ認証コード) アルゴリズムを指定します。 アルゴリズムを優先順に指定します。 既定値は「hmac-sha1,hmac-sha256,hmac-sha512,hmac-md5,hmac-ripemd160,hmac-sha1-96,hmac-md5-96」です。接続が ***FIPS モード*** 『22ページ 』で実行するように設定されている場合、既定値は「hmac-sha1,hmac-sha256,hmac-sha512」です。

-N

リモートコマンドを実行しません。転送するポートのみを構成する場合に便利です(プロトコルバージョン 2 のみ)。

-o *option*

構成ファイル 『118ページ 』で対応するオプションを指定します。例えば、次のように入力します。

```
ssh "-o FIPSMode=yes" myuser@myhost
```

-p *port*

サーバ上の接続先ポートを指定します。既定値 (22) は、Secure Shell 接続の標準ポートです。 これは、***構成ファイル*** 『19ページ 』にホスト単位で指定できます。

-q

クワイエットモードを有効にします。このモードでは、バナーを含むすべての警告および診断メッセージが表示されません。

-R *localport:remotehost:hostport*

(Secure Shell サーバを実行するコンピュータ上の) 指定されたリモートポートからのデータを、安全なトンネルを介して、指定された宛先ホストおよびポートにリダイレクトします。 詳細については、「リモートポート転送」を参照してください。 ポート転送を構成ファイルに指定することもできます。管理者としてログインしないかぎり、権限ポート (ポート番号 1024 以下) を転送できません。IPv6 アドレスは、別の構文 `port/host/hostport` を使用して指定できます。

-S

シェルを実行しないでください。

-t

コマンドが指定されている場合も TTY を強制的に割り当てます。

-T

pseudo-tty 割り当てを無効にします。

-v

デバッグレベルを冗長モードに設定します。これは、デバッグレベルを 2 に設定することと同じです。

-V

製品名およびバージョン情報を表示して終了します。コマンドラインで他のオプションが指定された場合、それらは無視されます。

-x

X11 接続の転送を無効にします。

-X

X11 接続の転送を有効にし、X11 クライアントを信頼されないものとして扱います。ゲストのリモート X11 クライアントは、信頼される X11 クライアントに属するデータを不正に変更できません。

X11 転送を有効にする場合は注意が必要です。ユーザの X 認可データベースのリモートホストでファイル権限を回避できるユーザは、転送された接続を介してローカル X11 ディスプレイにアクセスできます。攻撃者は、キーストローク監視などの行動を実行できる可能性があります。

-Y

X11 接続の転送を有効にし、X11 クライアントを信頼関係があるクライアントとして扱います。

X11 転送を有効にする場合は注意が必要です。ユーザの X 認可データベースのリモートホストでファイル権限を回避できるユーザは、転送された接続を介してローカル X11 ディスプレイにアクセスできます。攻撃者は、キーストローク監視などの行動を実行できる可能性があります。

-1

ssh でプロトコルバージョン 1 のみを試行します。プロトコルバージョン 1 は廃止される可能性があるため、推奨されません。

-2

ssh でプロトコルバージョン 2 のみを試行します。

-4

IPv4 アドレスのみを使用して接続させます。

-6

IPv6 アドレスのみを使用して接続させます。

ssh2 コマンドラインユーティリティ

構文: ssh2 [options] [user@]hostname [host command]

ssh2 コマンドラインユーティリティは、Secure Shell の F-Secure 実装と互換性があるスイッチに対応しています。

注意:

- F-Secure **ssh2** コマンドラインユーティリティを使用して記述されたスクリプトが存在する場合を除き、**ssh**『96ページ 』を使用する必要があります。**ssh2** *ではありません*。
- **ssh2** を使用する接続では、*既定のクライアント構成ファイル*『19ページ 』を使用しません。これらの接続では、(存在する場合) F-Secure 構成ファイルを使用します。

対応しているスイッチを表示するには、コマンドウィンドウに次のように入力にします。

```
ssh2 -h
```

ssh-keygen コマンドラインユーティリティ

ssh-keygen - クライアントおよびサーバ認証に使用されるキーの作成、管理、および変換。

一覧

```
ssh-keygen [-b bits] -t type [-N [passphrase]] [-C comment] [-f output_keyfile]
ssh-keygen -B [-f input_keyfile]
ssh-keygen -c [-P passphrase] [-C comment] [-f keyfile]
ssh-keygen -e [-f input_keyfile]
ssh-keygen -p [-P old_passphrase] [-N new_passphrase] [-f keyfile]
ssh-keygen -i [-f input_keyfile]
ssh-keygen -y [-f input_keyfile]
ssh-keygen -l [-f input_keyfile]
```

説明

ssh-keygen コマンドラインユーティリティを使用すると、公開鍵認証用の RSA 鍵と DSA 鍵の作成、既存の鍵のプロパティの編集、ファイル形式の変換が可能です。オプションを指定しないと、**ssh-keygen** では 2048 ビットの RSA 鍵のペアを生成し、秘密鍵を保護するために鍵の名前とパスフレーズの入力を求められます。公開鍵は秘密鍵と同じ名前を使用して作成され、拡張子 .pub が追加されます。鍵の生成が完了すると、鍵の場所が表示されます。

オプション

-b *bits*

鍵のサイズを指定します。鍵のサイズを大きくすると、ある程度までセキュリティは向上します。鍵のサイズを大きくすると最初の接続が遅くなりますが、正常に接続した後は、鍵のサイズはデータストリームの暗号化や解読の速度に影響しません。使用する鍵の長さは、多くの要素に依存します。その要素には、鍵の種類、鍵の有効期間、保護するデータの値、潜在的な攻撃者にとって利用可能なリソース、この非対称鍵とともに使用する対称鍵のサイズなどがあります。ニーズに合った最適な選択をするには、セキュリティ管理者にお問い合わせください。鍵のサイズは、64 ビットで均等に割り切れる次の値に切り上げられます。DSA 鍵の既定値は 1024 ビットで、RSA 鍵の既定値は 2048 ビットです。

-B

指定された鍵の指紋を、SHA-1 Bubble Babble 形式で表示します。 **-f** を使用して、鍵ファイルを指定できます。ファイルを指定していない場合、ファイル名の入力を求められます。 秘密鍵または公開鍵の名前を指定できますが、どちらの場合でも、公開鍵は使用可能でなくてはなりません。

-c

秘密鍵ファイルと公開鍵ファイルのコメントの変更を要求します。この操作は、RSA1 鍵にのみ対応します。秘密鍵を含むファイル、鍵にパスフレーズがある場合はパスフレーズ、新しいコメントの入力をプログラムにより求められます。

-C *comment*

鍵ファイル内のコメントフィールドに情報を指定します。文字列に空白が含まれている場合、引用符を使用します。鍵の作成時にコメントを指定していない場合、鍵の種類、作成者、日付、時刻を含む既定のコメントが作成されます。

-e

指定した OpenSSH 公開鍵または秘密鍵を使用して、Reflection 形式の公開鍵を生成します。 **-f** を使用して、鍵ファイルを指定できます。ファイルを指定していない場合、ファイル名の入力を求められます。

-f *filename*

生成される秘密鍵のファイル名を指定します(公開鍵も作成され、常に秘密鍵と同じ名前となり、.pub というファイル拡張子が付きます)。 このオプションを **-e**、**-i**、**-l**、**-p**、**-y**、および **-B** と組み合わせて、入力ファイル名の指定に使用することもできます。

-i

指定された Reflection 公開鍵を OpenSSH 形式に変換します。 **-f** を使用して、鍵ファイルを指定できます。ファイルを指定していない場合、ファイル名の入力を求められます。

-h

コマンドラインオプションに関する簡単な説明を表示します。

-l

MD5 ハッシュを使用し、指定された公開鍵ファイルの指紋を表示します。 **-f** を使用して、鍵ファイルを指定できます。ファイルを指定していない場合、ファイル名の入力を求められます。 秘密鍵を指定した場合、**ssh-keygen** では一致する公開鍵ファイルの検索を試行し、その指紋を印刷します。

-N *passphrase*

パスフレーズを設定します。例えば、新しい鍵のパスフレーズを指定するには、次のようにします。

```
ssh-keygen -N mypassphrase -f keyfile
```

パスフレーズ保護されていない新しい鍵を作成するには、次のようにします。

```
ssh-keygen -N -f keyfile
```

-N を **-p** および **-P** と組み合わせて使用すると、既存の鍵のパスフレーズを変更できます。

-p

このオプションを使用して、既存の秘密鍵のパスフレーズを変更します。このオプションのみを使用する場合、秘密鍵を含むファイル、古いパスフレーズ、新しいパスフレーズ (2 回) の入力をプログラムにより求められます。**-f**、**-P**、および **-N** と組み合わせて使用し、パスフレーズを非対話形式で変更できます。次に例を示します。

```
ssh-keygen -p -f keyfile -P oldpassphrase -N newpassphrase
```

-P *passphrase*

(古い) パスフレーズを指定します。

-q

ssh-keygen を終了します。

-t *type*

鍵の生成に使用する鍵のアルゴリズムを指定します。 プロトコルバージョン 2 に指定可能な値は「rsa」または「dsa」です。

-y

指定した秘密鍵を使用して、公開鍵の新しいコピーを取得します。 **-f** を使用して、鍵ファイルを指定できます。ファイルを指定していない場合、ファイル名の入力を求められます。

返される値

コマンドが正常に完了した場合、**ssh-keygen** は 0 (ゼロ) を返します。ゼロ以外の値は失敗を示します。

sftp コマンドラインユーティリティ

構文:sftp [*options*] [*user@*]*host*[*#port*]:*source_file*
[*user@*]*host*[*#port*][:*destination_file*]

注意: 既存の Secure Shell 接続を再利用できます。ただし、再利用するには、各コマンドラインでこれを明示的に有効にするか、SSHConnectionReUse 環境変数を Yes に設定する必要があります。詳細については、「*Secure Shell セッションにおける接続の再利用*『28ページ 』」を参照してください。

コマンドラインオプション

-a

ASCII モードでファイルを転送します。

-b *buffersize*

1 つの要求の最大バッファサイズを設定します。有効な値は 1024 ～ 32768 です。

-B *batchfile*

ログイン成功後、指定したバッチファイルで各コマンドを実行し、接続を終了します。例えば、以下のコマンドでは myname を使用して myhost に接続し、myfile でコマンドを実行します。ファイル内のすべてのコマンドを実行後、接続が終了します。

```
sftp -B c:\mypath\myfile myhost.com myname
```

バッチファイルでは、以下に示す対話型コマンドのいずれかを使用できます。

注意: セミコロンは、-B オプションを使って sftp コマンドに提供されたスクリプト内のコメントには使用できません。これらのバッチファイル内のコメントに印を付けるには、番号記号 (#) を使用します。

-c *cipher*

優先順に指定した暗号のカンマ区切りリスト。既定値は、「aes128-ctr,aes128-cbc,aes192-ctr,aes192-cbc,aes256-ctr,aes256-cbc,3des-cbc,blowfish-cbc,cast128-cbc,arcfour128,arcfour256,arcfour」です。接続が *FIPS モード*『22ページ 』で行われるように設定されている場合、既定値は「aes128-ctr,aes128-cbc,aes192-ctr,aes192-cbc,aes256-ctr,aes256-cbc,3des-cbc」です。

プロトコルバージョン 1 (廃止される可能性があるため、推奨されません) では、単一の暗号の指定が許可されています。対応する値は「3des」、「blowfish」、「des」です。

-C

すべての送信データの圧縮を有効にします。圧縮はモデム回線やほかの低速接続に適していますが、高速のネットワークでは応答速度の低下を招くだけです。

-d

ターゲットをディレクトリにします。

-F *config_file*

この接続に使用する代替構成ファイルを指定します。構成ファイルをコマンドラインで指定した場合、ほかの*構成ファイル*『19ページ 』は無視されます。

-h

コマンドラインオプションに関する簡単な説明を表示します。

-H *scheme*

この接続に使用する ***SSH 構成セクション*** 『116ページ 』を指定します。

-i *key_file*

鍵認証に使用する秘密鍵を指定します。 鍵ファイルは、ホスト単位で***構成ファイル*** 『19ページ 』に指定することもできます。複数の **-i** オプション (および構成ファイルに指定した複数の鍵) を指定できます。 ファイルまたはパスが空白を含む場合、引用符を使用します。

-k *directory*

config、ホスト鍵、ユーザ鍵ファイルの代替位置を指定します。注意:-k が使用されていると、既知のホストファイルが指定した場所にすでに存在していない場合のみ、その場所からホスト鍵の読み取りと書き込みが行われます。既知のホストファイルが見つからない場合、既定の場所で既知のホストファイルにホスト鍵の読み取りと書き込みが行われます。

-m *mac_spec*

この接続に使用する 1 つまたは複数のカンマ区切り MAC (メッセージ認証コード) アルゴリズムを指定します。 アルゴリズムを優先順に指定します。 既定値は「hmac-sha1,hmac-sha256,hmac-sha512,hmac-md5,hmac-ripemd160,hmac-sha1-96,hmac-md5-96」です。接続が ***FIPS モード*** 『22ページ 』で実行するように設定されている場合、既定値は「hmac-sha1,hmac-sha256,hmac-sha512」です。

-o *option*

構成ファイル 『118ページ 』で対応するオプションを指定します。例えば、次のように入力します。

```
ssh "-o FIPSMode=yes" myuser@myhost
```

-p

タイムスタンプとファイル属性を保持します。

-P *port*

リモートホストに接続するポート。

-q

クワイエットモードを有効にします。このモードでは、バナーを含むすべての警告および診断メッセージが表示されません。

-Q

進行状況のインジケータの表示をオフにします。

-R *maximum_requests*

同時要求の最大数を指定します。この数を増やすと、ファイル転送速度が多少向上しますが、メモリ使用量が増えます。既定値は未処理要求数 16 個です。

-s *subsystem*

ssh サブシステムを指定します。

-S *program*

暗号化された接続に使用するプログラム。

-u

コピー後、ソースファイルを削除します。

-v

デバッグレベルを冗長モードに設定します。これは、デバッグレベルを 2 に設定することと同じです。

-V

製品名およびバージョン情報を表示して終了します。コマンドラインで他のオプションが指定された場合、それらは無視されます。

-4

IPv4 アドレスのみを使用して接続させます。

-6

IPv6 アドレスのみを使用して接続させます。

対話型モード

ascii

転送の種類を ASCII に設定します。

binary

転送の種類をバイナリに設定します。

bye

sftp を終了します。

cd *path*

リモートディレクトリを path に変更します。

chmod *path*

path に関連付けられている許可を変更します。**mode** を使用して、3 桁の数値による許可を指定します。

lcd *path*

ローカルディレクトリを path に変更します。

exit

sftp を終了します。

get *remote-path* [*local-path*]

remote-path を取得し、ローカルマシンに保存します。ローカルパス名が指定されていない場合、リモートマシンと同じ名前が指定されます。

getext [*extension*,*extension...*]

ASCII 転送を使用するファイル拡張子を表示します。**setext** を使用して、この一覧を変更します。

help

ヘルプテキストを表示します。

lls [*ls-options* [*path*]]

path が指定されていない場合に、path または現在のディレクトリのローカルディレクトリ一覧を表示します。

lmkdir *path*

path で指定されたローカルディレクトリを作成します。

lpwd

ローカル作業ディレクトリを印刷します。

ls [*path*]

path が指定されていない場合に、path または現在のディレクトリのリモートディレクトリ一覧を表示します。

mkdir *path*

path で指定されたリモートディレクトリを作成します。

put *local-path* [*local-path*]

pwd

リモート作業ディレクトリを表示します。

quit

sftp を終了します。

reget *remote-file* [*local-file*]

指定した転送を再開します。これは **get** コマンドのように機能しますが、部分的に書き込まれたローカルファイルの存在を確認し、見つかった場合は最後に試行が中止された場所から転送を開始します。

rename *oldpath newpath*

リモートファイルの名前を oldpath から newpath に変更します。

rmdir *path*

path で指定されたリモートディレクトリを削除します。

rm *paths*

path で指定されたリモートファイルを削除します。

setext [*extension*,*extension...*]

ASCII 転送を使用するファイル拡張子を設定します。ワイルドカード文字を使用できます。引数が指定されていない場合、ファイル拡張子は ASCII 転送を使用しません。

version

sftp バージョンを表示します。

?

help と同じです。

sftp2 コマンドラインユーティリティ

構文: sftp [*options*] [*user@*]*host*[*#port*]:*source_file* [*user@*]*host*[*#port*][:*destination_file*]

sftp2 コマンドラインユーティリティは、Secure Shell の F-Secure 実装と互換性があるスイッチに対応しています。

注意:

- F-Secure **sftp2** コマンドラインユーティリティを使用して記述されたスクリプトが存在する場合を除き、**sftp**『104ページ 』を使用する必要があります。**sftp2** *ではありません*。
- **sftp2** を使用する接続では、*既定のクライアント構成ファイル*『19ページ 』を使用しません。これらの接続では、(存在する場合) F-Secure 構成ファイルを使用します。

対応しているスイッチを表示するには、コマンドウィンドウに次のように入力にします。

```
sftp2 -h
```

scp コマンドラインユーティリティ

構文: scp [options] [user@host:]file1 [user@host:]file2

scp コマンドラインユーティリティは、ネットワークのホスト間でファイルを安全にコピーします。データ転送に Secure Shell の **sftp** サブシステムを使用し、同じ認証を使用して Secure Shell と同じセキュリティを提供します。認証に必要な場合、**scp** からパスワードまたはパスフレーズの入力を求められます。ファイル名にホストとユーザの指定が含まれている場合があり、ホスト間でファイルをコピーすることを示します。

使用例

このコマンドラインは、ファイル f1 をホストからローカルマシンにコピーし、名前を f2 とします。

```
scp user@host:f1 f2
```

このコマンドは、ローカルファイル f1 をリモートホストの f2 にコピーします。

```
scp f1 user@host:f2
```

注意: 既存の Secure Shell 接続を再利用できます。ただし、再利用するには、各コマンドラインでこれを明示的に有効にするか、SSHConnectionReUse 環境変数を Yes に設定する必要があります。詳細については、「*Secure Shell セッションにおける接続の再利用*『28ページ 』」を参照してください。

オプション

次のオプションがあります。

-a

ASCII モードでファイルを転送します。

-b *buffersize*

1 つの要求の最大バッファサイズを設定します。

-B

バッチモードを設定すると、パスワードまたはパスフレーズの入力を求められません。パスフレーズのないユーザ鍵を使用して認証します。

-c *cipher*

優先順に指定した暗号のカンマ区切りリスト。既定値は、「aes128-ctr,aes128-cbc,aes192-ctr,aes192-cbc,aes256-ctr,aes256-cbc,3des-cbc,blowfish-cbc,cast128-cbc,arcfour128,arcfour256,arcfour」です。接続が ***FIPS モード*** 『22ページ 』で行われるように設定されている場合、既定値は「aes128-ctr,aes128-cbc,aes192-ctr,aes192-cbc,aes256-ctr,aes256-cbc,3des-cbc」です。

プロトコルバージョン 1 (廃止される可能性があるため、推奨されません) では、単一の暗号の指定が許可されています。対応する値は「3des」、「blowfish」、「des」です。

-C

圧縮を使用します。

-d

ターゲットをディレクトリにします。

-D *level*

デバッグレベルを設定します。指定可能な値は、1、2、および 3 です。

-F *configfile*

ユーザ単位の代替***構成ファイル*** 『19ページ 』を指定します。構成ファイルをコマンドラインで指定した場合、システム規模の構成ファイルは無視されます。

-h

コマンドラインオプションに関する簡単な説明を表示します。

-H *scheme*

この接続に使用する ***SSH 構成セクション*** 『116ページ 』を指定します。

-i *keyfile*

RSA 認証または DSA 認証の識別情報 (秘密鍵) を読み込むファイルを選択します。識別情報ファイルは、ホスト単位で構成ファイルに指定することもできます。複数の **-i** オプション (および***構成ファイル*** 『19ページ 』に指定した複数の識別情報) を指定できます。空白を含むパス名は、二重引用符で囲む必要があります。

-k *directory*

config、ホスト鍵、ユーザ鍵ファイルの代替位置を指定します。注意:-k が使用されていると、既知のホストファイルが指定した場所にすでに存在していない場合のみ、その場所からホスト鍵の読み取りと書き込みが行われます。既知のホストファイルが見つからない場合、既定の場所で既知のホストファイルにホスト鍵の読み取りと書き込みが行われます。

-l *limit*

帯域幅を指定した値 (Kb 単位) に制限します。

-o *option*

構成ファイル 『19ページ 』で使用される形式でオプションを指定するために使用できます。これは、別個のコマンドラインフラグがないオプションを指定する場合に便利です。対応するオプションの一覧については、「***構成ファイルのキーワード参照*** 『118ページ 』」を参照してください。

--overwrite

既存の転送先ファイルを上書きするかどうかを指定します。指定可能な値は「yes」および「no」です。既定値は「yes」です。

-p

タイムスタンプとファイル属性を保持します。

-P *port*

リモートホストに接続するポート。

-q

低度モード。バナーを含むすべての警告メッセージと診断メッセージを表示しないようにします。

-Q

進行状況のインジケータの表示をオフにします。

-r

すべてのサブディレクトリを含むディレクトリを再帰的にコピーします。

-u

コピー後、ソースファイルを削除します。

-v

高度モード。ssh により進捗状況に関するデバッグメッセージが表示されます。これは、接続、認証、構成に関する問題のデバッグに役立ちます。複数の **-v** オプションを使用すると、詳細度が高くなります。最大は 3 (-vvv) です。

-V

バージョン番号とアプリケーション情報を表示します。

-z

既定では、すべてのダウンロードについて、ファイル名の照合では大文字小文字が区別されます。このオプションを使用すると、ダウンロードで、サーバのファイル名指定にワイルドカードを使用した場合、大文字小文字は区別されません。

-1

プロトコルバージョン 1 のみにします。このオプションは、ssh トンネルを介して rcp を使用する OpenSSH サーバにもファイルを転送します。

-2

プロトコルバージョン 2 のみにします。

-4

IPv4 アドレスのみを使用します。

-6

IPv6 アドレスのみを使用します。

scp2 コマンドラインユーティリティ

scp2 ユーティリティは、F-Secure から移行中のお客様向けに提供されています。Secure Shell の F-Secure 実装と互換性があるスイッチに対応しています。

注意：

- F-Secure **scp2** コマンドラインユーティリティを使用して記述されたスクリプトが存在する場合を除き、**sftp**『108ページ 』または **sftp**『104ページ 』を使用する必要があります。**sftp2** *ではありません*。
- **scp2** を使用する接続では、*既定のクライアント構成ファイル*『19ページ 』を使用しません。これらの接続では、(存在する場合) F-Secure 構成ファイルを使用します。

対応しているスイッチを表示するには、コマンドウィンドウに次のように入力にします。

```
scp2 -h
```

付 録

付録

この章の内容

A 付 録

Secure Shell クライアントが使用するファイル

Reflection for Secure IT Secure Shell クライアントは、以下のファイルを使用します。

ユーザ固有の Secure Shell ファイル

これらのファイルは、現在 Windows にログインしているユーザ用の Secure Shell 接続に影響します。ファイルはユーザの .ssh フォルダにあります。

config

ユーザの構成ファイル。このファイルには、*SSH 構成セクション* 『116ページ 』で編成された Secure Shell 設定が含まれています。このファイルの内容は、*[Reflection Secure Shell の設定]* 『15ページ 』 ダイアログボックスで設定を変更するたびに更新されます。また、任意のテキストエディタを使用して、手動でこのファイルを編集することもできます。構成ファイルのキーワードリファレンス項目に、Reflection Secure Shell クライアントが対応しているキーワードの一覧が記載されています。

known_hosts

[Reflection Secure Shell 設定] ダイアログボックスの **[ホスト鍵]** 『40ページ 』 タブで **[信頼されるホスト鍵]** 一覧を更新したとき、または *[ホスト鍵認証]* 『41ページ 』 プロンプトで **[常時]** と答えたときに、Reflection は自動的にこのファイルを更新します。

システム全体の Secure Shell ファイル

これらのファイルは、コンピュータのすべてのユーザの Secure Shell 接続に影響します。これらのファイルは、手動で作成して *Reflection アプリケーションデータフォルダ* 『144ページ 』に配置する必要があります。

ssh_config

システム全体の構成ファイル。このファイルは、ユーザ構成ファイルに指定されていない値に対してマシン全体の既定値を提供します。

ssh_known_hosts

既知のホスト鍵のシステム全体の一覧。このファイルには、組織内のすべてのコンピュータの公開ホスト鍵を入れる必要があります。このファイルには、「システム名、公開鍵、およびオプションのコメントフィールド」の形式（フィールドはスペースで区切られます）で 1 行に 1 つの公開鍵が含まれています。同じコンピュータに異なる名前を使用するときは、このようなすべての名前をコンマで区切って一覧にする必要があります。ユーザがログインすると、クライアントホストを確認するために（ネームサーバから戻される）正規のシステム名が使用されます。Secure Shell はキーを確認する前にユーザから与えられた名前を正規の名前に変換しないため、その他の名前が必要です。これは、ネームサーバにアクセスできる人物がホスト認証を欺くことが可能であるためです。この一覧の鍵を表示できますが、[Reflection Secure Shell の設定] ダイアログボックスの *[ホスト鍵]* 『40ページ 』 タブにある **[グローバルホスト鍵]** 一覧で編集することはできません。

注意：構成ファイル内の **GlobalKnownHostsFile** *キーワード* 『118ページ 』を構成すると、ホスト鍵データベースの代替場所を構成できます。

PKI サポート用に Reflection が使用するファイル

これらのファイルは、PKI（公開鍵インフラストラクチャ）を使用して認証するように Reflection を構成した場合に使用されます。これらのファイルはユーザの .pki フォルダにあります。

pki_config

Reflection 証明書マネージャ 『49ページ』を使用して構成した設定が保存されます。これらの設定はすべての Reflection セッションで使用されます。

trust_store.p12

Reflection 証明書マネージャ 『49ページ』に追加される信頼されたルート証明書が含まれる PKCS#12 形式のファイル。

identity_store.p12

Reflection 証明書マネージャ 『49ページ』に追加される秘密鍵および証明書が含まれる、PKCS#12 形式のファイル。

cert_cache

中間ルート証明書のキャッシュ。このファイルを削除すると、キャッシュをクリアできます。

crl_cache

CRL (Certificate Revocation List) キャッシュ。このファイルを削除すると、キャッシュをクリアできます。

B 付 録

SSH 構成セクション

すべての Reflection Secure Shell 構成情報は、SSH 構成セクションを使用して *Secure Shell 構成ファイル*『19ページ 』に保存されます。 Secure Shell 接続時、Reflection では現在の SSH 構成セクションを使用して、接続方法を決定します。 また、設定を変更すると、Reflection では変更を現在の SSH 構成セクションに保存します。

特定のホストに固有の Secure Shell 設定を構成したい場合、SSH 構成セクション名をホスト名と同じにする必要があります。

注意: セクションを指定せずに **[Reflection Secure Shell の設定]** ダイアログボックスを開いた場合、Secure Shell 設定のいずれかを変更するとすぐに、現在指定されているホスト名を使用して、Reflection により新しい SSH 構成セクションが自動的に作成されます。

複数のホストに対する接続に同じ Secure Shell 設定を使用したい場合、**[Reflection Secure Shell の設定]** ダイアログボックスを開く前に、SSH 構成セクションに内容を表す名前を入力し、このセクションに保存したい設定を構成します。 セクションを作成および構成すると、以降のホストセッションの構成時にこのセクションを指定できます。

SSH 構成セクションの名前は、大文字と小文字が区別されます。

SSH 構成セクションの保存方法

Reflection Secure Shell 構成情報は、*Secure Shell 構成ファイル*『19ページ 』に保存されます。 SSH 構成セクション名は、キーワード **Host** を使用して識別されます。 構成ファイルは、**[Reflection Secure Shell の設定]** ダイアログボックスを閉じると更新されます。 構成する既定以外のすべての設定は、現在のセクションに保存されます。

例については、「*サンプル構成ファイル*『117ページ 』」を参照してください。

C 付 録

サンプル構成ファイル

このサンプル Secure Shell 構成ファイルには、2 つの SSH 構成セクション MyHost.Demo.com および GeneralSSH があります。

MyHost.Demo.com の設定では、実際のホスト名を使用する一連の Secure Shell 設定を指定します。 この設定は SSH 構成セクションとして MyHost.Demo.com を指定するすべての接続に使用され、SSH 構成セクションが指定されていない場合にそのホストへの接続にも使用されます。

GeneralSSH では実際のホストアドレスを指定しないため、この設定はセッション構成時にこの SSH 構成セクションを指定した場合にのみ使用されます。

この config ファイルで、新しいホスト (MyHost.Demo.com 以外) への接続を構成し、GeneralSSH セクションを指定しない場合、Reflection では既定の Secure Shell 設定を使用して接続します。

```
Host MyHost.Demo.Com
   Protocol 2
   KbdInteractiveAuthentication no
   ChallengeResponseAuthentication no
   PasswordAuthentication no
   RSAAuthentication no
   IdentityFile "C:¥SSHusers¥Joe¥.ssh¥mykey"
   LogLevel VERBOSE
#EndHost

Host GeneralSSH
   StrictHostKeyChecking yes
   ServerAlive yes
#EndHost
```

D 付 録

構成ファイルのキーワード参照 - Secure Shell 設定

Secure Shell *構成ファイル*『19ページ 』を手動で編集する場合に、この参照を使用します。構成ファイルは、キーワード **Host** によってそれぞれ識別されるセクションに分かれています。各セクションでは、指定したホストまたは *SSH 構成セクション*『116ページ 』を使用する、すべての接続に使用される Secure Shell 設定を指定します。

構成ファイルは、キーワードの後に値が続きます。構成オプションは、空白またはオプションの空白と 1 つの等号 (=) で区切ることができます。キーワードは大文字と小文字を区別しませんが、引数は大文字と小文字を区別します。

番号記号 (#) で始まる行はコメントです。空の行は無視されます。

注意:この一覧の項目では、Secure Shell 接続に影響する機能を構成します。その他のキーワードは、**ssh** コマンドラインセッションの端末エミュレーションの構成に使用できます。これらのキーワードに関する詳しい説明は、「*構成ファイルのキーワード参照 - 端末エミュレーション設定*『132ページ 』」を参照してください。

AddAuthKeyToAgent

この設定は、**ForwardAgent** が「yes」に設定されている場合に、クライアントの公開鍵認証の処理方法に影響します。サーバの公開鍵認証が成功し、**ForwardAgent** と **AddAuthKeyToAgent** の両方が「yes」に設定されている場合、認証に使用された鍵や証明書は Reflection 鍵エージェントに自動的に追加されます。この鍵は鍵エージェントには保存されませんが、鍵エージェントが実行されている間は使用できます。AddAuthKeyToAgent が「no」(既定値) に設定されている場合、鍵と証明書が鍵エージェントに自動的に追加されることはありません。鍵エージェントは、手動でインポートされた鍵のみを使用します。

AuthUseAllKeys

この設定は、クライアントの公開鍵認証の処理方法に影響します。この設定が「no」(既定値) の場合、クライアントは **IdentityFile** というキーワードを使用して指定した 1 つまたは複数の鍵のみを使用して認証しようとします。この設定が「yes」の場合、クライアントは使用できる公開鍵全部を使用して認証しようとします。

BatchMode

スクリプトおよびバッチジョブに使用可能なパスワードメッセージおよびパスフレーズメッセージを含む、ユーザ入力要求をすべて無効にするかどうかを指定します。 指定可能な値は「yes」および「no」です。既定値は「no」です。

注意:このキーワードは、キーボード対話型の認証構成時のユーザ入力要求を無効にしませんが、**BatchMode** 有効時にキーボード対話型を使用する接続は失敗します。

BindAddress

複数インタフェースまたは別名アドレスを使用して、コンピュータから送信するインタフェースを指定します。

ChallengeResponseAuthentication

試行/応答認証を使用するかどうかを指定します。引数は「yes」または「no」にする必要があります。この認証方式は、SecurID、PAM 認証、またはサーバからのメッセージとユーザからの応答が必要なその他の外部認証方式を使用している場合に推奨されます。既定値は「yes」です。これは、対応している SSH プロトコル 1 のみに適用されますが、推奨されません。SSH プロトコルバージョン 2 には **KbdInteractiveAuthentication** を使用します。

CheckHostIP

このフラグを「yes」に設定した場合、Reflection Secure Shell クライアントは、ホストの公開鍵のほか、known_hosts ファイルのホスト IP アドレスを確認します。既知のホスト一覧のホスト IP が、接続に使用している IP アドレスに一致する場合にのみ、接続が許可されます。既定値は「no」です。注意 **:StrictHostKeyChecking** が「no」の場合、この設定は適用されません。

CheckHostPort

このフラグを「yes」に設定した場合、Reflection Secure Shell クライアントは、ホストの公開鍵のほか、known_hosts ファイルのホストポートを確認します。既知のホスト一覧のホストポートが、接続に使用しているポートに一致する場合にのみ、接続が許可されます。既定値は「no」です。注意 **:StrictHostKeyChecking** が「no」の場合、この設定は適用されません。

Cipher

プロトコルバージョン 1 のセッションの暗号化に使用する暗号を指定します。現在、「blowfish」、「3des」、「des」に対応しています。des は、3des 暗号に対応していない従来のプロトコル 1 実装との相互運用性のため Secure Shell クライアントのみが対応しています。暗号上の弱点があるため、使用することは推奨されません。既定値は「3des」です。

Ciphers

プロトコルバージョン 2 に使用可能な暗号を優先順に指定します。複数の暗号は、カンマで区切る必要があります。既定値は、「aes128-ctr,aes128-cbc,aes192-ctr,aes192-cbc,aes256-ctr,aes256-cbc,3des-cbc,blowfish-cbc,cast128-cbc,arcfour128,arcfour256,arcfour」です。接続が FIPS モードで行われるように設定されている場合、既定値は「aes128-ctr,aes128-cbc,aes192-ctr,aes192-cbc,aes256-ctr,aes256-cbc,3des-cbc」です。

ClearAllForwardings

ローカル転送、リモート転送、または動的転送されたポートのうち、既に処理されたすべてのポートを構成ファイルまたはコマンドラインからクリアします。注意:**scp** および **sftp** は、この設定に関係なく、すべての転送ポートを自動的にクリアします。 指定可能な値は「yes」および「no」です。既定値は「no」です。

Compression

圧縮を有効にするかどうかを指定します。 圧縮は、モデム回線などの低速接続には向いていますが、高速ネットワークでは応答速度を低下させます。また、圧縮はパケットをより不規則にするため、悪意のある人物がパケットを解読することが難しくなります。 指定可能な値は「yes」および「no」です。既定値は「no」です。

CompressionLevel

圧縮を有効にした場合に使用する圧縮レベルを指定します。このオプションは、プロトコルバージョン 1 にのみ適用されます。引数は、1（高速）から 9（低速、最適）までの整数にする必要があります。既定レベルは 6 で、ほとんどのアプリケーションに有効です。値の意味は gzip と同じです。

ConnectionAttempts

終了前に試行する回数（1 秒間に 1 回）を指定します。引数は整数にする必要があります。これは、接続に失敗することがある場合にスクリプトで使用できます。既定値は「1」です。

ConnectionReuse

同一ホストへの複数のセッションで元の Secure Shell 接続を再利用するかどうかを指定します。このため、再認証が不要となります。引数は「yes」または「no」にする必要があります。「yes」に設定すると、ホスト名、ユーザ名、SSH 構成セクション（使用する場合）がすべて一致する場合、新しい接続は既存のトンネルを再利用します。「no」に設定すると、Reflection ではセッションごとに新しい接続を確立します。つまり、新しい接続ごとに認証処理を繰り返し、変更された接続固有の設定（転送および暗号など）も適用されます。Reflection ウィンドウを使用した接続で接続する場合、既定値は「yes」になります。接続に Reflection *コマンドラインユーティリティ* 『95ページ 』を使用している場合は「no」になります。詳細については、「*Secure Shell セッションにおける接続の再利用* 『28ページ 』」を参照してください。

ConnectTimeout

サーバへの接続完了を試行している時にクライアントが待機する最大時間（秒単位）を指定します。タイマーは、接続の確立時（ログオン前）に開始して、設定、ホスト鍵交換、および認証中に稼働します。実際には、タイマーは基本的に認証処理の間に稼働します。既定値は「120」です。

DisableCRL

ホスト証明書の検証時に CRL (Certificate Revocation List) の確認を行うかどうかを指定します。これを「yes」に設定すると、CRL の確認が無効になります。この設定の既定値は、CRL の確認に関する現在のシステム設定に従います。システム設定を表示して編集するには、Internet Explorer を起動して、**[ツール] - [インターネットオプション] - [詳細設定]** コマンドに進みます。**[セキュリティ]** の下の [サーバー証明書の取り消しを確認する] を探します。

DynamicForward

安全なチャネルを介して転送するローカルマシンの TCP/IP ポートを指定します。アプリケーションプロトコルは、リモートマシンと接続する場所を決定するために使用されます。引数はポート番号にする必要があります。現在、SOCKS4 プロトコルに対応し、Reflection Secure Shell は SOCKS4 サーバとして動作します。複数転送を指定できるため、コマンドラインで追加転送を指定できます。管理権限のあるユーザだけが権限ポートを転送できます。

EscapeChar

エスケープ文字を設定します (既定は'~')。エスケープ文字は、コマンドラインでも設定できます。引数は単一の文字「^」で、後に文字を続ける必要があります。あるいは、エスケープ文字を無効にするには「none」にする必要があります (バイナリデータ接続を透過にするため)。

FipsMode

この設定を「yes」にすると、アメリカ合衆国政府の連邦情報処理規格 (FIPS) 140-2 に合致するセキュリティプロトコルおよびアルゴリズムを使用して、接続する必要があります。この規格に合致しないオプションは、[暗号化] タブで使用できません。

注意：この設定はキーワード `Host` で指定される SSH 構成セクションに影響し、同じ SSH 構成セクション（またはホスト名）を使用するように構成されていないかぎり、以降の Secure Shell セッションに影響しません。

ForwardAgent

これを「yes」に設定すると、Reflection 鍵エージェント接続の転送が有効になります。エージェント転送を有効にする場合は注意が必要です。エージェントの UNIX ドメインソケットのリモートホストでファイル権限を回避できるユーザは、転送された接続を介してローカルエージェントにアクセスできます。攻撃者は鍵の情報を取得できませんが、エージェントに読み込まれた識別情報を使用して、その鍵で操作を実行して認証を有効にすることができます。サーバでもエージェント転送が有効になっている必要があります。既定値は「no」です。

ForwardX11

安全なチャネルを介して X11 接続を自動的にリダイレクトし、DISPLAY を設定するかどうかを指定します。引数は「yes」または「no」にする必要があります。既定値は「no」です。(注意:Reflection X を使用して Secure Shell を構成する場合は、「**ForwardX11ReflectionX**」を参照してください)。

ForwardX11ReflectionX

Reflection X (14.1 以降) 用の Secure Shell 接続を構成している場合にのみ、この設定を使用してください。安全なチャネルを介して X11 接続を自動的にリダイレクトし、DISPLAY を設定するかどうかを指定します。引数は「yes」または「no」にする必要があります。既定値は「yes」です。

GatewayPorts

リモートホストが、ローカル転送ポートへの接続を許可されるかどうかを指定します。既定では、Reflection Secure Shell はローカルポート転送をループバックアドレスに結合します。これによって、ほかのリモートホストが、転送ポートに接続できないようにしています。**GatewayPorts** を使用して、Reflection Secure Shell がローカルポート転送をワイルドカードアドレスに結合するように指定できます。これによって、リモートホストは転送ポートに接続できます。この設定を有効にする場合は注意が必要です。この設定によって、リモートホストで、認証なしにシステムで転送されたポートを使用できるようになるため、ネットワークと接続の安全性が低下します。引数は「yes」または「no」にする必要があります。既定値は「no」です。

GlobalKnownHostsFile

Reflection アプリケーションデータフォルダ 『144ページ 』の ssh_known_hosts という既定ファイルの代わりに使用する、グローバルホスト鍵データベース用のファイルを指定します。

注意 : パスまたはファイル名の一部に空白が含まれている場合、ファイル名を引用符で囲みます。

GssapiAuthentication

GSSAPI 認証を使用して、Kerberos KDC を認証するかどうかを指定します。この設定は、使用中のプロトコルがプロトコルバージョン 2 である場合にのみ適用できます(プロトコルバージョン 1 で同等の設定は **KerberosAuthentication** になります)。 指定可能な値は「yes」および「no」です。既定値は「no」です。

GssapiDelegateCredentials

GSSAPI を使用して、ホストに発券許可チケット (krbtgt) を転送するかどうかを指定します。この設定は、使用中のプロトコルがプロトコルバージョン 2 である場合にのみ適用できます(プロトコルバージョン 1 で同等の設定は **KerberosTgtPassing** になります)。 指定可能な値は「yes」および「no」です。既定値は「yes」です。

GssapiUseSSPI

Microsoft の Security Support Provider Interface (SSPI) を GSSAPI 認証に使用するかどうかを指定します。この設定は、Kerberos/GSSAPI 認証が有効な場合にのみ使用できます (プロトコルバージョン 2 では **GssapiAuthentication** を使用、プロトコルバージョン 1 では **KerberosAuthentication** を使用)。このキーワードの引数は「yes」または「no」にする必要があります。「no」に設定すると、Reflection Secure Shell クライアントは GSSAPI 認証に Reflection Kerberos クライアントを使用します。「yes」に設定すると、Reflection Secure Shell クライアントは Secure Shell サーバに対する認証に Windows ドメインのログイン資格情報 (SSPI) を使用します。SSPI はプロトコルバージョン 2 接続のみに対応しており、サーバが **GSSAPI-with-mic** 認証方式に対応する必要があります。既定値は「yes」です。

GssServicePrincipal

クライアントが Kerberos Key Distribution Center (KDC) へサービスチケット要求を送信する時に、使用する既定以外のサービスプリンシパル名を指定します。GSSAPI プロバイダとして SSPI を選択している場合、この設定を使用して Windows ドメインとは異なるレルムにサービスプリンシパルを指定できます。完全なホスト名の後に @ とレルム名を続けます。例えば、`myhost.myrealm.com@MYREALM.COM` のようになります(既定で、ホスト名の値は接続する Secure Shell サーバの名前になり、レルムは **GssapiUseSSPI** の値によって異なります。**GssapiUseSSPI** が「no」の場合、レルム名は既定のプリンシパルプロファイルで指定されます。**GssapiUseSSPI** が「yes」の場合、レルムは Windows ドメイン名になります)。

Host

指定した *SSH 構成セクション* 『116ページ 』に属する (次の **Host** キーワードまで) 続く宣言を指定します。文字「*」および「?」は、ワイルドカードとして使用できます。パターンに単一の「*」を使用すると、すべてのホストにグローバルな既定設定を指定できます。Reflection 接続では、最初に一致する **Host** 文字列 (ワイルドカード文字を含む) を使用します。以降の一致は無視されます。

注意:[Reflection Secure Shell の設定] ダイアログボックスを閉じる時、既定の設定値は構成ファイルに保存されません。既定値を手動でファイルに追加している場合、ダイアログボックスを閉じる時にその既定値は削除されます。ワイルドカードホストのスタンザを、特定のホスト名を使用するスタンザと組み合わせて使用する場合、これにより、設計制約が課せられます。ワイルドカードのスタンザで構成された値を上書きするよう設定された特定のホストのスタンザに既定値を手動で構成した場合、ホスト固有の SSH 構成セクションの設定を表示するために [Reflection Secure Shell の設定] ダイアログボックスを開くと、既定値は削除されます。この状況は、グローバル構成ファイルを使用すると適切に処理することができます。グローバル構成ファイルは、ユーザが [Reflection Secure Shell の設定] ダイアログボックスを開いたり閉じたりしても、更新されません。

HostKeyAlgorithms

クライアントが使用するホスト鍵アルゴリズムを優先順に指定します。このオプションの規定は次のとおりです。「x509v3-rsa2048-sha256, x509v3-sign-rsa, x509v3-sign-dss, ssh-rsa-sha2-256@attachmate.com, ssh-rsa,ssh-dss」この設定は、証明書と標準ホスト鍵認証の両方をサーバに構成する場合に便利です。既定値では、通常の SSH 鍵アルゴリズムの前に x509 アルゴリズムが指定されています。SSH プロトコルでは、ホストの認証試行は 1 回しか許可しません。(これは、複数の認証試行に対応しているユーザ認証とは異なります)。ホストが証明書を提示し、証明書を使用したホスト認証がクライアントに構成されていない場合、x509 アルゴリズムが優先されていると接続は失敗しますこの場合、優先順位を「ssh-rsa-sha2-256@attachmate.com, ssh-rsa,ssh-dss, x509v3-rsa2048-sha256, x509v3-sign-rsa, x509v3-sign-dss」に変更して、証明書よりも SSH 鍵が優先されるようにクライアントを構成します。

HostKeyAlias

ホスト鍵データベースファイルでホスト鍵の検索または保存のため、実際のホスト名の代わりに使用する別名を指定します。このオプションは、ssh 接続のトンネリングまたは単一のホストで複数のサーバを実行している場合に使用できます。

IdentityFile

鍵認証に使用する秘密鍵を指定します。 (ファイルはユーザの .ssh フォルダにあります)。IdentityFile 項目は、**[Reflection Secure Shell の設定]** ダイアログボックスの **[ユーザ鍵]** タブの一覧から鍵または証明書を選択すると追加されます。構成ファイルに複数の識別情報ファイルを指定でき、これらのすべての識別情報が順番に試行されます。

注意:空白が含まれている場合、完全なパス名を引用符で囲みます。

KbdInteractiveAuthentication

キーボード対話型認証を使用するかどうかを指定します。 指定可能な値は「yes」および「no」です。既定値は「yes」です。 この認証方式は、SecurID、PAM 認証、またはサーバからのメッセージとユーザからの応答が必要なその他の外部認証方式を使用している場合に推奨されます。パスワード有効期限または最初のログインパスワードの変更が有効になっているホストのパスワード認証で、PasswordAuthentication 方式よりも有効に機能する場合があります。認証に成功するために有効期限が切れたパスワードをリセットする必要がある場合、パスワード認証に必要な場合もあります。これは、SSH プロトコル 2 にのみ適用されます。SSH プロトコルバージョン 1 には ChallengeResponseAuthentication を使用します。

KeepAlive

システムが TCP キープアライブメッセージを相手に送信する必要があるかどうかを指定します。送信されると、接続の切断またはいずれかのマシンのクラッシュが検出されます。ネットワークのダウンまたはリモートホストの停止をクライアントが検出するため、既定値は「yes」(キープアライブを送信する) になっています。これはスクリプトにとって重要であり、ユーザにとっても役に立ちます。ただし、ルートが一時的にダウンした場合に接続が切断されることになるので、一部のユーザにとっては迷惑かもしれません。キープアライブを無効にするには、値を「no」に設定します。このキーワードにより、Windows TCP キープアライブ設定が有効になり、既定では 2 時間ごとにキープアライブメッセージが送信されます。TCP/IP キープアライブは、Windows レジストリには通常存在しない 2 つのオプションパラメータ(KeepAliveTime および KeepAliveInterval) を使用して構成できます。これらは、以下の場所の HKEY_LOCAL_MACHINE レジストリサブツリーに構成されます。

```
SYSTEM\CurrentControlSet\Services\Tcpip\Parameters
```

これらのパラメータの設定については、Microsoft Knowledge Base Article 120642 を参照してください。

KerberosAuthentication

Kerberos 認証をプロトコルバージョン 1 接続に使用するかどうかを指定します(プロトコルバージョン 2 で同等の設定は **GssapiAuthentication** になります)。このキーワードの引数は「yes」または「no」にする必要があります。

KerberosTgtPassing

Kerberos TGT をサーバに転送するかどうかを指定します。これは、Kerberos サーバが実際に AFS kaserver の場合にのみ機能します。この設定は、プロトコルバージョン 1 にのみ適用されます(プロトコルバージョン 2 で同等の設定は **GssapiDelegateCredentials** になります)。このキーワードの引数は「yes」または「no」にする必要があります。

KexAlgorithms

クライアントが対応する鍵交換アルゴリズムと優先順位を指定します。対応する値は、「diffie-hellman-group1-sha1」、「diffie-hellman-group-exchange-sha1」、および「diffie-hellman-group14-sha1」です。既定値は、「diffie-hellman-group1-sha1,diffie-hellman-group-exchange-sha1,diffie-hellman-group14-sha1」です。場合によっては、「diffie-hellman-group14-sha1」を他の 2 つよりも前に配置するために、鍵交換アルゴリズムの順番を変更する必要があります。hmac-sha512 MAC を使用する場合、または鍵交換中に次のエラーが表示される場合は、この操作が必要です。"fatal: dh_gen_key: group too small:1024 (2*need 1024)"

注意:Reflection Kerberos クライアントを使用する GSSAPI 認証が有効の場合、追加の鍵交換アルゴリズム (gss-group1-sha1 および gss-gex-sha1) が一覧に自動的に追加されます。

LocalForward

安全なチャネルを介して、リモートマシンの指定したホストとポートに転送するローカルマシンの TCP/IP ポートを指定します。複数の転送を指定できます。管理権限のあるユーザだけが権限ポートを転送できます。FTP の転送、リモートデスクトップの構成、接続後の実行ファイル (*.exe) の自動起動にオプションの引数を構成することもできます。このキーワードの構文は次のとおりです。

```
LocalForward localport host:hostport [FTP=0|1] [RDP=0|1] ["ExecutableFile" [args]]
```

次のオプションがあります。

localport	ローカルポート番号。
host:*hostport*	リモートホストおよびそのホストのポート(localhost を指定して、すでに Secure Shell 接続を確立した同じリモートホストの異なるポートにデータを転送できます)。IPv6 アドレスは、別の構文 host/port を使用して指定できます。
FTP	FTP ファイル転送をトンネル接続する場合、1 に設定します。
RDP	リモートデスクトップセッションをトンネル接続する場合、1 に設定します。

"*ExecutableFile*"	Secure Shell 接続確立後すぐに Reflection がアプリケーションを起動するように、実行ファイル（必要に応じて完全なパス情報を含む）を指定します。安全なトンネルを介してデータを転送するには、指定した *localport* を使用して *localhost*（またはループバック IP アドレス、127.0.0.1）に接続するようにこのアプリケーションを構成する必要があります。

Logfile

デバッグに使用するログファイルを指定します。すべてのセッションの入力および出力は、このファイルに書き込まれます。次に示すように、このキーワードと **-o** コマンドラインユーティリティオプションを使用します。

```
-o Logfile=\path\logfile_name
```

注意：パスまたはファイル名の一部に空白が含まれている場合、パスとファイル名を引用符で囲みます。

LogLevel

Reflection Secure Shell クライアントからのメッセージの記録時に使用する詳細レベルを指定します。指定可能な値は、QUIET, FATAL, ERROR, INFO, VERBOSE, DEBUG, DEBUG1, DEBUG2 および DEBUG3 です。既定値は「INFO」です。DEBUG と DEBUG1 は同じです。DEBUG2 と DEBUG3 はそれぞれ、より高いレベルの詳細出力を指定します。

Macs

MAC（メッセージ認証コード）アルゴリズムを優先順に指定します。MAC アルゴリズムは、データ整合性保護のためにプロトコルバージョン 2 で使用されます。複数のアルゴリズムは、カンマで区切る必要があります。既定の場所は以下のとおりです。"hmac-sha256, hmac-sha2-256, hmac-sha1, hmac-md5, hmac-ripemd160, hmac-sha1-96, hmac-md5-96, hmac-sha512, hmac-sha2-512"接続が FIPS モードで行われるように設定されている場合、既定値は「hmac-sha256, hmac-sha2-256, hmac-sha1, hmac-sha512, hmac-sha2-512」です。

MatchHostName

ホストの証明書の検証時にホスト名の一致を確認するかどうかを指定します。この設定が「yes」の場合（既定値）、Reflection で構成したホスト名が、証明書の CommonName フィールドまたは SubjectAltName フィールドに入力されているホスト名に一致していなければなりません。

Multihop

一連の SSH サーバによって安全な接続を確立するために使用可能な ***マルチホップ*** 『67ページ』接続を構成します。これは、直接リモートサーバにアクセスすることはできないが、中間サーバを介してアクセスすることができるネットワーク構成で役に立ちます。

このキーワードの構文は次のとおりです。

```
Multihop localport host:hostport ["SSH config scheme"]
```

一連の各サーバに **Multihop** 行を新しく追加します。一覧の各接続は、その上の接続で確立されたトンネルを介して送信されます。

下記の例では、サーバ C に構成された SSH 接続が最初に サーバ A に接続し、それから サーバ B、最後に サーバ C に接続します。

```
Host ServerC
  Multihop 2022 ServerA:22
  Multihop 3022 ServerB:22
```

オプションで *SSH 構成セクション* 『116ページ 』を指定して、チェーン内の任意のホストの Secure Shell 設定を構成できます。例えば、次のように入力します。

```
Multihop 4022 joe@ServerA:22 "Multihop SchemeA"
```

NoShell

NoShell を「Yes」に設定すると、クライアントは端末セッションを開かずにトンネルを作成します。このオプションは ConnectionReuse と組み合わせて使用し、その他の ssh 接続が再利用できるトンネルを作成できます。注意:このオプションは、接続がコマンドラインユーティリティで行われた場合に適用されます。Reflection for Secure IT ユーザインタフェースでの使用には対応していません。

NumberOfPasswordPrompts

失効するまでに試行できるパスワードメッセージの入力回数を指定します。このキーワードの引数は整数にする必要があります。既定値は「3」です。

PasswordAuthentication

パスワード認証を使用するかどうかを指定します。 指定可能な値は「yes」および「no」です。既定値は「yes」です。

Port

リモートホストに接続するポート番号を指定します。既定値は「22」です。

PreferredAuthentications

クライアントがプロトコル 2 認証方式を試行する順番を指定します。これは、**[Reflection Secure Shell の設定]** ダイアログボックスの **[全般]** タブの **[ユーザ認証]** 一覧に表示される方式の順番（上から下）に対応します。この設定によって、クライアントはある方式（キーボード対話型など）を別の方式（パスワードなど）より優先させることができます。既定で、Reflection は「publickey,keyboardinteractive,password」の順番で認証を試行します。GSSAPI 認証が有効な場合、既定値は「gssapi-with-mic,external-keyex,gssapi,publickey,keyboard-interactive,password」に変更されます。

注意：

- config ファイルに **PreferredAuthentications** が含まれている場合、指定する一覧に試行したい認証方式をすべて含める必要があります。**PreferredAuthentications** が存在するにもかかわらず特定の認証方式を指定しない場合、その認証方式を有効にするキーワードが正しく構成されていても、Reflection ではその認証方式を使用しません。
- **PreferredAuthentications** 一覧に認証方式を含めることによって、その方式を使用する認証が有効になるわけではありません。既定で使用されない認証を有効にするには、その認証方式のキーワードも正しく構成する必要があります（例えば、GSSAPI 認証を有効にするには、**GssapiAuthentication** を yes に設定する必要があります）。

PreserveTimestamps

サーバ間でファイルを転送する時に、属性とタイムスタンプを変更するかどうかを指定します。このキーワードが「no」(既定値) の場合、タイムスタンプと属性が変更されます。「yes」の場合、ファイルの元のタイムスタンプと属性が保持されます。

Protocol

Reflection Secure Shell クライアントが対応するプロトコルバージョンを優先順に指定します。指定可能な値は「1」および「2」です。複数の値は、カンマで区切る必要があります。既定値は「2,1」で、Reflection はバージョン 2 を試行し、バージョン 2 が使用できない場合にバージョン 1 にフォールバックします。

Proxy

Secure Shell 接続に使用するプロキシの種類を指定します。対応する値は、「SOCKS」と「HTTP」です。

注意：この設定を使用する各 Host セクションで、プロキシの使用が有効になります。プロキシサーバアドレスは、ユーザごとに Windows レジストリに保存されます。

PubkeyAuthentication

公開鍵認証を試行するかどうかを指定します。このオプションは、プロトコルバージョン 2 にのみ適用されます。 指定可能な値は「yes」および「no」です。既定値は「yes」です。

RemoteCommand

リモートサーバで実行する 1 つまたは複数のコマンドを指定します。UNIX サーバへの接続時には、セミコロン (;) を使用して複数のコマンドを区切ります。Windows サーバへの接続時には、アンパサンド記号 (&) を使用してコマンドを区切ります。接続の確立後、サーバは指定したコマンドを実行 (または実行を試行) し、セッションが終了します。サーバは、クライアントから受信したコマンドの実行を許可するよう構成されている必要があります。

コマンドは、使用するサーバに合った構文を使用して指定する必要があります。例えば、以下が同じになります。

UNIX の場合:`ls ; ls -l`

Windows の場合:`dir/w & dir`

RemoteForward

安全なチャネルを介して、ローカルマシンの指定したホストとポートに転送するリモートマシンの TCP/IP ポートを指定します。最初の引数はポート番号で、2 番目の引数は *host*:*port* にする必要があります。IPv6 アドレスは、別の構文 *host/port* を使用して指定できます。複数転送を指定できます。管理権限のあるユーザだけが権限ポートを転送できます。

RSAAuthentication

RSA 認証を試行するかどうかを指定します。このオプションは、プロトコルバージョン 1 にのみ適用されます。RSA 認証は、識別情報ファイルが存在する場合のみ試行されます。 指定可能な値は「yes」および「no」です。既定値は「yes」です。

SendEnv

シェルまたはコマンドの実行前に、サーバに設定する環境変数を指定します。値の形式は `VAR val` にする必要があります。サーバは指定した変数に対応し、これらの環境変数を受け付けるように構成する必要があります。

ServerAlive

ServerAliveInterval で指定した間隔で、SSH サーバにサーバアライブメッセージを送信するかどうかを指定します。Secure Shell の ServerAlive 設定では、SSH プロトコルメッセージを指定した間隔でサーバに送信し、サーバが機能していることを確認します。この設定が有効になっていないと、サーバが停止するかネットワーク接続が失われた場合に SSH 接続が終了されません。この設定は、TCP セッションのみを転送する接続がサーバによって時間切れになるのを防ぐためにも使用できます。サーバは、SSH トラフィックが存在しないことの検出を理由にこれらの接続を時間切れにする場合があるからです。 指定可能な値は「yes」および「no」です。既定値は「no」です。

注意:Secure Shell の ServerAlive 設定は、すべての TCP/IP 接続がファイアウォールによって時間切れになるのを防ぐために Windows レジストリに設定可能な TCP キープアライブ設定 (KeepAlive) とは関係ありません。TCP/IP キープアライブの動作を変更するには、Windows レジストリを編集する必要があります。

ServerAliveInterval

ServerAlive = 'yes' の場合に使用する間隔（秒単位）を指定します。1 以上の整数値を使用します。既定値は「30」です。

SftpBufferLen

SFTP 転送時に、各パケットで要求されるバイト数を指定します。既定値は「32768」です。この値を調整すると、転送速度を上げることができます。最適な値は、使用しているネットワークおよびサーバ設定によって異なります。この値を変更すると、データの転送をキャンセルしてから実際に転送が停止するまでの時間にも影響を与えることがあります。

SftpMaxRequests

クライアントが SFTP 転送時に許可する、未処理データ要求の最大数を指定します。既定値は「10」です。この値を調整すると、転送速度を上げることができます。最適な値は、使用しているネットワークおよびサーバ設定によって異なります。この値を変更すると、データの転送をキャンセルしてから実際に転送が停止するまでの時間にも影響を与えることがあります。

SftpVersion

クライアントが SFTP 接続に使用するバージョンを指定します。有効な値は 3 と 4 です。この設定が 4 (既定値) の場合、接続では (サーバが対応していれば) SFTP バージョン 4 が使用されます。また、サーバがバージョン 4 に対応していない場合はバージョン 3 が使用されます。この設定が 3 の場合、クライアントは常に SFTP バージョン 3 を使用します。注意:SFTP 4 が必要な場合は、Attachmate 技術サポートに問い合わせて、ご使用の Reflection クライアントのバージョンでこの設定が使用できることを確認してください。

StrictHostKeyChecking

引数は「yes」、「no」または「ask」にする必要があります。既定値は「ask」です。このオプションを「yes」に設定した場合、Reflection Secure Shell クライアントはホスト鍵を known_hosts ファイル（ユーザの .ssh フォルダにある）に自動的に追加せず、ホスト鍵が変更されたホストへの接続を拒否します。このオプションを使用する場合、ユーザは新しいホストを手動で追加する必要があります。このフラグを「no」に設定した場合、Reflection は確認ダイアログボックスを表示せずにホストに接続し、ホスト鍵を信頼する鍵の一覧に追加しません。このフラグを「ask」に設定した場合、ユーザが該当する鍵であることを確認してから、新しいホスト鍵がユーザ既知のホストファイルに追加されます。すべての場合、既知のホストのホスト鍵は自動的に検証されます。

注意：この設定は、ホストが x509 証明書を使用する認証を構成されている場合は影響しません。ホストがホスト認証のために証明書を提示し、トラストアンカとして構成された必要な CA 証明書がない場合、接続に失敗します。

TryEmptyPassword

このフラグを「yes」に設定した場合、クライアントは空のパスワードの入力を試行してパスワード認証を開始します。これは、ほとんどのシステムでログイン試行と見なされます。

User

ログインするユーザを指定します。これは、異なるマシンで異なるユーザ名を使用する場合に便利です。

UseOCSP

クライアントが OCSP (Online Certificate Status Protocol) を使用してホスト証明書を検証するかどうかを指定します。 指定可能な値は「yes」および「no」です。既定値は「no」です。

UserKeyCertLast

Reflection クライアントが公開鍵認証中に証明書の署名を処理する方法を指定します。この設定が「yes」(既定) の場合、クライアントは最初に標準の ssh 鍵署名 (ssh-rsa または ssh-dss) を使用して証明書を送信します。それに失敗すると、クライアントは再び証明書署名 (x509-sign-rsa または x509-sign-dss) の使用を試みます。場合によっては、この 2 番目の試みが行われず、認証が失敗します。この設定が「no」の場合、クライアントは最初に証明書署名を試み、次に ssh 鍵署名を試みます。

UserKnownHostsFile

known_hosts ファイル (ユーザの .ssh フォルダにある) の代わりに使用する、ユーザホスト鍵データベース用のファイルを指定します。 ファイルまたはパスが空白を含む場合、引用符を使用します。

x509dsasigtype

DSA 秘密鍵の所有を提供する過程でクライアントが使用するハッシュアルゴリズムを指定します。指定可能な値は「sha1raw」(既定値) と「sha1asn1」です。

x509rsasigtype

RSA 秘密鍵の所有を提供する過程でクライアントが使用するハッシュアルゴリズムを指定します。指定可能な値は「md5」、「sha1」(既定値)、および「sha256md5」です。

X11Display

X11 転送が有効な場合、X11 プロトコルが通信する PC のローカルループバックインタフェース上のポートを転送するかどうかを決定します。

注意:Reflection X (バージョン 12.x、13.x、または 14.x) を使用している場合は、このキーワードを構成する必要はありません。Reflection X サーバと Reflection Secure Shell クライアントは自動的に同期して、X サーバ表示設定 (［設定］ - ［表示］ - X 表示番号) に基づく適切なポートを使用します。この場合、X11Display キーワードは無視されます。別の PC X サーバを使用する場合は、このキーワードを使用して、PC X サーバに定義されている適切なリスニングポートを指定します。

既定値は、[0] です。これにより、ポート 6000 への転送が構成されます。このポートは、X11 プロトコル規約で定義された既定のリスニングポートです。指定する表示値は実際のリスニングポートを決定するために、6000 に追加されます。例えば、**X11Display** を 20 に設定した場合、Secure Shell クライアントにとって、PC-X サーバがポート 6020 で待ち受けしていることを意味します。

E 付 録

構成ファイルのキーワード参照 - 端末エミュレーション設定

この一覧の項目は、Reflection *ssh*『96ページ 』コマンドラインセッションと *ssh2*『100ページ 』コマンドラインセッションの端末エミュレーション設定を構成します。これらの設定は、Secure Shell *構成ファイル*『19ページ 』に手動で追加するか、コマンドラインで -o スイッチを使用して実行できます。

注意:これらの設定は、コマンドライン端末セッションにのみ適用されます。Reflection ユーザインタフェースで実行される端末セッションには適用されません。

構成ファイルは、キーワード **Host** によってそれぞれ識別されるセクションに分かれています。各セクションでは、指定したホストまたは *SSH 構成セクション*『116ページ 』を使用する、すべての接続に使用される設定を指定します。

構成ファイルは、キーワードの後に値が続きます。構成オプションは、空白またはオプションの空白と 1 つの等号 (=) で区切ることができます。キーワードは大文字と小文字を区別しませんが、引数は大文字と小文字を区別します。

番号記号 (#) で始まる行はコメントです。空の行は無視されます。

引用符は、空白を含む文字列引数の前後に必要になります。端末エミュレーションのキーワードおよび引数は、大文字と小文字を区別しません。

注意:Secure Shell 接続を構成するためのキーワードは、別個の一覧に示されています。「*構成ファイルのキーワード参照 - Secure Shell 設定*『118ページ 』」を参照してください。

AnswerBackMessage

AutoAnswerback を「yes」に設定すると、AnswerBackMessage でアンサバック要求に応じてホストに送信される文字列を指定します (ENQ 文字 ASCII 5)。

指定可能な文字列値は次のとおりです。30 字までの文字列値。
既定値は"" (ヌル文字列) です。
サンプル構文は次のとおりです。

```
AutoAnswerback yes
AnswerbackMessage "My answer back string"
```

AutoAnswerback

AutoAnswerback を「yes」に設定すると、キーワード AnswerBackMessage を使用して指定したメッセージ文字列が接続後にホストに自動的に送信されます。

指定可能な値は「yes」または「no」です。
既定値は「no」です。
サンプル構文は次のとおりです。

```
AutoAnswerback yes
AnswerbackMessage "My answer back string"
```

AutoWrap

カーソルが右余白に到達した時の動作を指定します。「yes」に設定すると、カーソルが端末ウィンドウの右余白に到達した時に、文字が次の行に自動的に折り返されます。「no」に設定すると、カーソルが右余白に到達した時に自動的に進みません。追加の文字を入力すると、各文字はカーソルを移動するまで前の文字を上書きします。

指定可能な値は「yes」または「no」です。
既定値は「no」です。
サンプル構文は次のとおりです。

```
AutoWrap yes
```

BackspaceKeyIsDel

バックスペースキーの動作を指定します。「no」に設定すると、バックスペースキーはバックスペース（ASCII 8）文字を送信します。「yes」に設定すると、バックスペースキーは削除（ASCII 127）文字を送信します。

指定可能な値は「yes」または「no」です。
既定値は「no」です。
サンプル構文は次のとおりです。

```
BackspaceKeyIsDel yes
```

CursorKeyMode

クライアントがカーソルキーパッドのキーをどうのように処理するかを指定します。「no」に設定すると、カーソルキーパッドは通常モードに設定されます。つまり、カーソルキーパッドのキーはカーソルエスケープシーケンスを送信します。「yes」に設定すると、カーソルキーパッドはアプリケーションモードに設定されます。つまり、カーソルキーパッドのキーはアプリケーションエスケープシーケンスを送信します。

指定可能な値は「yes」または「no」です。
既定値は「no」です。
サンプル構文は次のとおりです。

```
CursorKeyMode yes
```

CursorStyle

カーソルスタイルを指定します。

指定可能な文字列値は次のとおりです。Block、Blockblink、Line、Lineblink
既定値は「Lineblink」です。
サンプル構文は次のとおりです。

```
CursorStyle Block
```

CursorVisible

カーソルを表示するかどうかを指定します。「no」に設定すると、カーソルは端末ウィンドウに表示されません。

指定可能な値は「yes」または「no」です。
既定値は「yes」です。
サンプル構文は次のとおりです。

```
CursorVisible no
```

DisplayCols

端末ウィンドウの桁数を設定します。

指定可能な値は次のとおりです。最小値は「80」です。使用できる最大値は、モニターサイズと表示設定によって異なります。
既定値は現在のコマンドウィンドウのサイズによって決まります。
サンプル構文は次のとおりです。

```
DisplayCols 120
```

DisplayRows

端末ウィンドウの行数を設定します。

指定可能な値は次のとおりです。最小値は「24」です。使用できる最大値は、モニターサイズと表示設定によって異なります。
既定値は現在のコマンドウィンドウのサイズによって決まります。
サンプル構文は次のとおりです。

```
DisplayRows 30
```

HostCharacterSet

既定以外のホスト文字セットを指定します。

指定可能な文字列値は次のとおりです。

```
PC437_English                 Windows1256
PC737_Greek                   Windows1257
PC775_Baltic                  Windows1258
PC850_Multilingual            Korean_Johab
PC852_Slavic                  ISOLatin_1
PC855_Cyrillic                ISOLatin_2
PC857_Turkish                 ISOLatin_3
PC858_Multilingual_Euro       ISO_Baltic
PC860_Portuguese              ISO_Cyrillic
PC861_Icelandic               ISO_Arabic
PC862_Hebrew                  ISO_Greek
PC863_CanadianFrench          ISO_Hebrew
PC864_Arabic                  ISOLatin_5
PC865_Nordic                  ISOLatin_9
PC866_Cyrillic                ISO2022_JIS
PC869_ModernGreek             ISO2022_JIS-Allow
PC932_Shift_JIS               ISO2022_JIS-X0201_1989
PC936_SimplifiedChinese       ISO2022_Korean
PC949_Korean                  ISO2022_SimplifiedChinese
PC950_TraditionalChinese      ISO2022_TraditionalChinese
DECMultinational              EUC_Japanese
UCS2                          EUC_SimplifiedChinese
Windows1250                   EUC_Korean
Windows1251                   EUC_TraditionalChinese
Windows1252                   GB2312_SimplifiedChinese
Windows1253                   GB18030_SimplifiedChinese
Windows1254                   UTF7
Windows1255                   UTF8
```

既定値は「PC437_English」です。
サンプル構文は次のとおりです。

```
HostCharacterSet EUC_Japanese
```

InsertMode

入力を挿入モードにするか置換モードにするかを指定します。「no」に設定すると、入力によりカーソル位置の既存の文字が置換されます。「yes」に設定すると、新しい文字がカーソル位置に挿入され、既存の文字は右に移動します。

指定可能な値は「yes」または「no」です。
既定値は「no」です。
サンプル構文は次のとおりです。

```
InsertMode yes
```

InverseVideo

端末ウィンドウが反転表示を使用するかどうかを指定します。「yes」に設定すると、すべての画面属性の前景色および背景色が反転されます。

指定可能な値は「yes」または「no」です。
既定値は「no」です。
サンプル構文は次のとおりです。

```
InverseVideo yes
```

KeyBoardActionMode

キーボードを使用可能にするかどうかを指定します。「yes」に設定すると、キーボードがロックされ使用できません。

指定可能な値は「yes」または「no」です。
既定値は「no」です。
サンプル構文は次のとおりです。

```
KeyBoardActionMode yes
```

MarginBell

マージンベルを鳴らすかどうかを指定します。「yes」に設定すると、カーソルが右余白から 8 文字の時にベルが鳴ります。この設定を「no」に設定すると、マージンベルが鳴りません。

指定可能な値は「yes」または「no」です。
既定値は「yes」です。
サンプル構文は次のとおりです。

```
MarginBell no
```

NewLine

クライアントを行送りモードにするか改行モードにするかを指定します。「no」(行送りモード) に設定すると、[Enter] キーを押すと復帰のみが送信されます。行送り、改ページ、垂直タブを受信すると、カーソルが現在の桁の 1 つ下の行に移動します。「yes」(改行モード) に設定すると、[Enter] キーを押した時に復帰と行送りの両方が送信されます。改ページ、垂直タブを受信すると、カーソルが次の行の先頭桁に移動します。

指定可能な値は「yes」または「no」です。
既定値は「no」です。
サンプル構文は次のとおりです。

```
NewLine yes
```

NRCSet

対応する文字列値のいずれかを使用して、異なる国別文字セットを指定します。これを有効にするには、キーワード **UseNRC** も「yes」に設定する必要があります。

指定可能な文字列値は次のとおりです。

```
British                 Norwegian
Finnish                 Portuguese
French                  EuropeanSpanish
CanadianFrench          Swedish
German                  SwissGerman
Italian
```

既定値は「ASCII」です。
サンプル構文は次のとおりです。

```
UseNRC yes
```

```
NRCSet British
```

NumericKeyPadMode

クライアントが数字キーパッドのキーをどのように処理するかを指定します。「yes」に設定すると、キーパッドは数字モードに設定されます。つまり、キーパッドのキーを押すと数値が送信されます。「no」に設定すると、キーパッドはアプリケーションモードに設定されます。つまり、キーパッドのキーはホーム、上、右などのアプリケーションエスケープシーケンスを送信します。

指定可能な値は「yes」または「no」です。
既定値は「no」です。
サンプル構文は次のとおりです。

```
NumericKeyPadMode no
```

OriginMode

カーソルのホーム位置を指定します。「no」に設定すると、カーソルのホーム位置は端末ウィンドウの左上隅になります。「yes」に設定すると、カーソルのホーム位置は端末ウィンドウの余白設定に関連して決まります。

指定可能な値は「yes」または「no」です。
既定値は「no」です。
サンプル構文は次のとおりです。

```
OriginMode yes
```

SevenBitControls

8 ビット C1 制御コードを送信するかどうかを指定します。「yes」に設定すると、8 ビット C1 制御コードに相当する 7 ビット制御コードが送信されます。「no」に設定すると、8 ビット C1 制御コードが送信されます。

注意：ssh コマンドラインクライアントの HostCharacterSet の既定値は PC437_English です。C1 制御を送信したい場合、HostCharacterSet を DECMultinational または ISOLatin 文字セットのいずれかに設定する必要があります。

指定可能な値は「yes」または「no」です。
既定値は「yes」です。
サンプル構文は次のとおりです。

```
SevenBitControls no
```

TerminalModel

クライアントがエミュレートする端末の種類を指定します。

指定可能な文字列値は次のとおりです。vt52、vt102、vt220
既定値は「vt220」です。
サンプル構文は次のとおりです。

```
TerminalModel vt102
```

UseNRC

これを「yes」に設定すると、キーワード NRCSet を使用して国別文字セットを指定できます。

指定可能な値は「yes」または「no」です。
既定値は「no」です。
サンプル構文は次のとおりです。

```
UseNRC yes
NRCSet British
```

UseANSIColor

「yes」に設定すると、ANSI 色エスケープシーケンスに対応します。

指定可能な値は「yes」または「no」です。
既定値は「yes」です。
サンプル構文は次のとおりです。

```
UseANSIColor no
```

WarningBell

警告ベルを鳴らすかどうかを指定します。「yes」に設定すると、ホストからベル文字（ASCII 7）を受信したり、キーボードから入力されたりすると、ベルが鳴ります。この設定を「no」に設定すると、警告ベルは鳴りません。

指定可能な値は「yes」または「no」です。
既定値は「yes」です。
サンプル構文は次のとおりです。

```
WarningBell no
```

F 付 録

DOD PKI 情報

このセクションでは、Reflection をインストール、構成、使用して DOD (Department of Defense) 環境またはその他の PKI (Public Key Infrastructure) 環境内で操作する方法について解説します。PKI 構成は、Secure Shell 接続および SSL/TLS 接続の両方に影響します。

DOD PKI モードでの Reflection の実行

既定では、Reflection アプリケーションにより DOD PKI 要件を満たさない構成が一部許可されます。管理者は、Reflection グループポリシを使用して DOD PKI 要件を満たすようにすべての Reflection セッションを構成できます。

DOD PKI モードを構成するには

1 次のいずれかの方法で、グループポリシエディタを実行します。

- コマンドラインで Gpedit.msc
  -or- と入力します。
- [Active Directory ユーザとコンピュータ] から、[組織単位] のプロパティを開き、**[グループポリシ]** タブをクリックして、ポリシオブジェクトを編集または新規作成します。

2 *Reflection テンプレート (ReflectionPolicy.adm) がインストールされていない場合はインストールします。*『93ページ 』

Reflection ポリシテンプレートのダウンロードおよびインストールの方法については、「*技術ノート 2216『http://support.attachmate.com/techdocs/2216.html』*」を参照してください。

---

3 **[ローカルコンピュータポリシ]** - **[ユーザの構成]** - **[管理用テンプレート]** - **[Reflection の設定]** で、**[DoD PKI 以外のモードを許可する]** を無効にします。

DOD PKI モードの構成には、以下の影響があります。

- *CRL*『143ページ 』を確認したり、*OCSP*『143ページ 』レスポンダを使用するように Reflection を構成する必要があります。DOD PKI モードでは、いずれの確認形式も使用しないオプションは無効です(SSH 接続の場合、[Reflection Secure Shell の設定] ダイアログボックスの [PKI] タブを使用して、証明書の取り消しを構成します。SSL/TLS 接続の場合、[PKI の構成] ダイアログボックスを使用して構成します)。
- Reflection では、FIPS 承認の暗号化アルゴリズムを実行します。SSH 接続の場合、[Reflection Secure Shell の設定] ダイアログボックスの [暗号化] タブでは FIPS 承認オプションのみ使用できます。SSL/TLS 接続の場合、**[暗号化レベル]** を 40 ビットまたは 56 ビットに設定できません。
- 接続を確立するためには、証明書のホスト名が Reflection 接続に指定したホスト名と完全に一致している必要があります。つまり、**[証明書のホスト名と対象ホスト名が一致するかどうかを確認する]** が自動的にオンになり、変更することはできません(SSH 接続の場合、**[Reflection Secure Shell の設定]** ダイアログボックスの **[PKI]**『48ページ 』タブを使用して、この設定を構成します。SSL/TLS 接続の場合、**[PKI の構成]** ダイアログボックスを使用して構成します)。
- MD2 または MD5 ハッシュを使用して署名された中間 CA 証明書は、証明書検査に対応していません。

トラストポイントのインストールおよび削除

トラストポイントは、信頼チェーン内の任意の *CA*『143ページ 』証明書です。

トラストポイントを Reflection 証明書格納場所に追加するには

1 *[Reflection 証明書マネージャ] を開きます*『49ページ 』。

2 **[信頼された認証局]** タブをクリックします。

3 **[インポート]** をクリックしてから、証明書 (通常、*.cer または *.crt) を検索して指定します。

トラストポイントを Reflection 証明書格納場所から削除するには

1 *[Reflection 証明書マネージャ] を開きます*『49ページ 』。

2 **[信頼された認証局]** タブをクリックします。

3 証明書を選択して、**[削除]** をクリックします。

注意

- 中間 CA トラストポイントは、LDAP サーバまたは HTTP サーバから取得できます。このサーバは、証明書の AIA (Authority Information Access) 拡張に定義されている明示的な URI、または [Reflection 証明書マネージャ] の [LDAP] タブに構成されている LDAP サーバ情報を使用して特定できます。これらの証明書は、<マイドキュメント>¥Attachmate¥Reflection¥.pki または ¥All users¥Application data¥Attachmate¥Reflection にある cert_cache ファイルに保存されます。
- Reflection が DOD PKI モードで実行中の場合、[Reflection 証明書マネージャ] に追加したルート証明書のみが使用されます。Windows の証明書格納場所に存在する可能性がある DOD PKI 以外の証明書を削除する必要はありません。

証明書取り消しの確認の構成

Reflection では、証明書取り消しの確認の既定値は現在のシステム設定に基づいて決まります。システムが CRL の確認を行うように構成されている場合は、既定で Reflection セッションにおいて *CRL*『143ページ 』を使用して証明書取り消しが確認されます。OCSP レスポンダを使用するように Reflection を構成することもできます。

Reflection では、CRL 確認を無効にする設定にも対応しています。この設定をテストに使用できますが、Reflection が DOD PKI モードで実行中の場合にはこのオプションを使用できません。

警告:CRL 確認を無効にすると、セキュリティ上のリスクが高まります。このオプションはテストにのみ使用します。

中間証明書または CRL を取得する 1 台または複数の LDAP サーバを定義できます。

LDAP サーバを定義するには

1 *[Reflection 証明書マネージャ] を開きます。*『49ページ 』

2 **[LDAP]** タブをクリックします。

3 **[追加]** をクリックし、次の URL 形式を使用してサーバを指定します。

```
ldap://hostname:portnumber
```

例えば、次のように入力します。

```
ldap://ldapserver.myhost.com:389
```

OCSP を構成するには

1 証明書の取り消し情報を要求する 1 台または複数の OCSP サーバを定義できます。

2 **[証明書失効の確認]** を **[OCSP を使用する]** に設定します(SSH 接続の場合、[Reflection Secure Shell の設定] ダイアログボックスの [PKI] タブを使用します。SSL/TLS 接続の場合、[PKI の構成] ダイアログボックスを使用します)。

証明書で必要な OCSP レスポンダの URL は、証明書の AIA 拡張に指定されます。この情報が証明書で提供されない場合、次の手順を使用して OCSP レスポンダ情報を構成できます。

---

3 *[Reflection 証明書マネージャ] を開きます。*『49ページ 』

4 **[OCSP]** タブをクリックします。

5 **[追加]** をクリックし、次の URL 形式を使用してサーバを指定します。

```
URL:portnumber
```

例えば、次のように入力します。

```
https://ocspmachine.host.com:389
```

DOD PKI サービスの URI の使用

Reflection では、*CRL*『143ページ 』の自動更新および取得に URI を使用できます。RFC3280 のセクション 4.2.1.14 に定義されています。

CRL 確認が有効の場合、Reflection では以下のように証明書の取り消しを確認します。

1. crl_cache ファイルで有効な取り消し情報を確認します。見つからない場合は、手順 2 に進みます。
2. 証明書の CDP 拡張で HTTP URI または LDAP URI を確認し、指定した順番 (最初に HTTP、次に LDAP) で問い合わせます。取り消し対象の証明書が見つかった場合は、接続を切断します。証明書が見つからない場合は、手順 3 に進みます。
3. 1 台または複数の LDAP サーバが [Reflection 証明書マネージャ] の [LDAP] タブに指定されている場合、証明書の発行者の拡張子に示されている CA の識別名をまとめて、CRL ファイルに問い合わせます。いずれの CRL にも取り消し対象の証明書が見つからない場合は、次の検証手順に進みます。

期限切れの CRL の更新は自動的に処理されるため、管理者の介入または構成の必要はありません。

OCSP 確認が有効の場合、Reflection ではすべての使用可能な OCSP レスポンダを必ず確認します。これは、証明書が取り消されたことをいずれかのレスポンダが把握している場合に、接続が失敗することを確認するためです。接続を確立するためには、少なくとも 1 つの OCSP レスポンダが使用可能であり、認証ステータスに対して値「good」を返す必要があります。
Reflection では、以下のように確認を実行します。

1. 証明書の AIA 拡張で 1 つまたは複数の OCSP レスポンダを確認し、各レスポンダに問い合わせます。いずれかのレスポンダからの証明書のステータスが「revoked」に戻った場合、接続を切断します。
2. [Reflection 証明書マネージャ] の [OCSP] タブを使用して指定した 1 つまたは複数のユーザ構成 OCSP レスポンダを確認し、各レスポンダに問い合わせます。いずれかのレスポンダからの証明書のステータスが「revoked」に戻った場合、接続を切断します。
3. すべてのレスポンダが「unknown」を返した場合、接続を切断します。少なくとも 1 つの OCSP レスポンダから「good」応答が返された場合、次の検証手順に進みます。

URI を使用した中間証明書の取得

RFC3280 のセクション 4.2.2.1 に定義されているように、Reflection では以下のように URI を使用して中間 *CA*『143ページ』証明書を取得できます。

1. cert_cache ファイルで必要な中間証明書を確認します。見つからない場合は、手順 2 に進みます。
2. HTTP URI または LDAP URI のいずれかが証明書の AIA (Authority Information Access) 拡張に定義されている場合、中間 CA 証明書の取得にこれらの使用を試行します (最初に HTTP、次に LDAP)。
3. 前の試行に失敗した場合、発行している証明書の件名から識別名をまとめて、CACertificate 属性の内容に定義された LDAP サーバに問い合わせます。

Reflection では証明書のセキュリティポリシ拡張を実施するため、セキュリティポリシ構成は不要です。

証明書と秘密鍵の構成と保護

証明書を使用してクライアント認証を構成するには

1 *[Reflection 証明書マネージャ] を開きます。*『49ページ』

2 **[個人]** タブで **[インポート]** をクリックしてから、証明書 (通常、*.pfx または *.p12) を検索して指定します。この鍵を使用するごとに要求されるパスフレーズを作成する画面が表示されます。システムでこの鍵を保護するのに役立つため、パスフレーズの入力が推奨されます。

3 Secure Shell 接続の場合、**[Reflection Secure Shell の設定]** ダイアログボックスを開き、**[ユーザ鍵]** タブをクリックして、現在指定しているホストへのクライアント認証に使用したい証明書を選択します (このステップは、SSL/TLS 接続には必要ありません)。

秘密鍵の保護

クライアントの秘密鍵が盗まれた場合、悪意のあるユーザがそのユーザにアクセス可能な任意のサーバのファイルにアクセスできます。このリスクを最小限にするには、各クライアントユーザがパスフレーズを使用して自分の秘密鍵を必ず保護する必要があります。これによって、パスフレーズを知っている人だけがその鍵で認証できることになります。ユーザは、組織のセキュリティポリシのパスワード仕様に従ってパスフレーズを作成し、保護する必要があります。

鍵が改ざんされた場合の操作

秘密鍵が不正な人物によって利用可能になった場合、または鍵にアクセスする人物の操作を信用しない理由がある場合、秘密鍵が改ざんされたと見なします。

クライアントの鍵が改ざんされた場合、クライアントの証明書を取り消します。

改ざんされた鍵を置換するには

1 新しい秘密鍵と証明書を生成して、[Reflection 証明書マネージャ] に鍵をインポートします。

2 識別情報が変更された場合、サーバでこのクライアントの割り当てファイルの行を更新します。

クライアントコンピュータから改ざんされた鍵を削除するには

1 [Reflection 証明書マネージャ] の **[個人]**『49ページ』タブから鍵を削除します。これによって、identity_store.p12 ファイルからこの鍵が削除されます。

2 古い鍵と証明書を含む元のファイル (*.pfx or *.p12) がまだクライアントコンピュータにある場合は、DOD 承認ファイル削除ユーティリティを使用してこのファイルを削除します。

用語集

C

CA (認証局)

信頼される組織にある、電子証明書を発行するサーバ。CA は、新しい証明書の発行を管理し、認証に対して有効でなくなった証明書を取り消します。CA は証明書発行権限を 1 つまたは複数の中間 CA に委任して、信頼のチェーンを形成することがあります。最高レベルの CA 証明書は「信頼されたルート」とみなされます。

CRL (Certificate Revocation List)

認証局によって失効された、電子署名された証明書の一覧。CRL で識別された証明書はすでに有効ではありません。

G

GSSAPI (Generic Security Services アプリケーションプログラムインタフェース)

プログラムにセキュリティサービスへのアクセスを提供するアプリケーションプログラミングインタフェースです。

K

Kerberos

信頼されたサードパーティを使用して TCP/IP ネットワーク上で安全な通信を実現するプロトコル。このプロトコルは、プレーンテキストのパスワードではなく、暗号化されたチケットを使用してより安全にネットワーク認証を行います。

M

MAC (メッセージ認証コード)

データが送信中に変更されていないことの確認に使用されます。MAC は、データと共有秘密鍵が含まれる任意の長さのパケットを使用して作成されたハッシュです。送信側と受信側は、共有鍵および合意したアルゴリズムを使用して、転送されたデータの各パケットの MAC を独自に計算します。メッセージが送信中に変更されている場合、別のハッシュ値になり、パケットは拒否されます。

P

PKCS

PKCS (Public Key Cryptography Standards: 公開鍵暗号標準) の略。RSA 研究所によって考案および公布された、公開鍵暗号の実装間の互換性を確保可能な一覧の標準。各 PKCS 標準では、特定の暗号化用途の仕様が定められています。Reflection for Secure IT Windows クライアント は、次の PKCS 標準を使用しています。

- PKCS#11 は、スマートカードや USB トークンのようなハードウェアデバイスを使用する認証サポートを提供しています。

- PKCS#12 は、証明書および関連する秘密鍵の保管と送信に使用されます。この形式のファイルの拡張子は、通常、*.pfx または *.p12 です。Reflection for Secure IT は、この形式で格納された証明書と鍵を使用して認証をサポートしています。

R

Reflection アプリケーションデータフォルダ

Reflection は、すべてのユーザが利用できる Secure Shell 情報を次の場所に格納します。

Windows XP、Windows Server 2003 の場合:
¥Documents and Settings¥all users¥Application Data\Attachmate\Reflection

Windows 7、Windows Vista、Windows Server 2008 の場合:
¥ProgramData\Attachmate\Reflection

S

Secure Shell

リモートコンピュータへのログインとコマンドの実行を安全に行うためのプロトコル。これは、Telnet、FTP、rlogin、あるいは rsh の代わりとなる安全な方法です。Secure Shell 接続では、ホスト（サーバ）とユーザ（クライアント）の両方の認証が必要です。また、ホスト間の通信はすべて暗号化された通信チャネルを介して行う必要があります。また、Secure Shell では X11 セッションまたは指定の TCP/IP ポートを、安全なトンネルを介して転送することもできます。

U

UTC (Universal Time, Coordinated; 協定世界時)

高精度の時間標準。時間帯を記述する場合、UTC は、グリニッジ子午線（経度 0）での時刻であるグリニッジ*標準時*を表します。一般に、UTC 時間は 24 時間制で与えられます。

W

Windows ユーザプロファイルフォルダ

ユーザプロファイルフォルダは、Windows システム管理者が構成できます。既定は次の通りです。

- Windows7、Windows Server 2008:
  \Users*username*\
- Windows Server 2003:
  \Documents and Settings*username*\

Windows 共通アプリケーションデータフォルダ

注意：アプリケーションデータフォルダは、既定では表示されません。

既定の場所は以下のとおりです。

- Windows 7、Windows Vista、Windows Server 2008 の場合:
  ¥ProgramData¥
- Windows XP、Windows Server 2003 の場合:
  ¥Documents and Settings¥all users¥Application Data¥

データの整合性

データが元のソースから変更されていない保証です。データの整合性を維持する方法は、データが誤って、または悪意を持って変更、改ざん、破壊されていないことを保証するように設計されています。

デジタル署名

送信されたメッセージの信頼性と整合性の確認に使用されます。通常、送信者は公開/秘密鍵のペアのうち秘密鍵を保有し、受信者は公開鍵を保有します。署名を作成するには、送信者はメッセージからハッシュを計算し、この値を自らの秘密鍵で暗号化します。受信者は、送信者の公開鍵を使用して署名を復号化し、受信したメッセージのハッシュを独自に計算します。復号化した値と計算した値が一致した場合、受信者は、送信者が秘密鍵の保有者であり、メッセージが送信中に改ざんされていないことを信頼します。

パスフレーズ

パスフレーズはパスワードに類似していますが、一連の語句、句読点、数字、空白、任意の文字列を組み合わせたフレーズを使用できる点が違います。パスフレーズは、秘密鍵や鍵エージェントなどの保護されたオブジェクトへのアクセスを制限して、セキュリティを向上させます。

ハッシュ

「メッセージダイジェスト」と呼ぶこともあり、ハッシュまたはハッシュの値は可変長のデジタルデータから生成される固定長の数値です。ハッシュは元のデータよりもかなり小さく、計算によって生成されますが、別のデータから同一のハッシュを作成することは統計的にできないようになっています。

ポート転送

安全でないトラフィックを安全な SSH トンネルを介してリダイレクトする方法です。ポート転送には、ローカルとリモートの 2 種類があります。ローカルポート転送（発信ポート転送）は、指定されたローカルポートから送信された発信データを、安全なチャネルを介して、指定されたリモートポートに送信します。クライアントアプリケーションとサーバとの間でデータを安全に交換するには、関連サーバを実行するコンピュータにクライアントを直接接続するのではなく、リダイレクトされるポートに接続するように構成します。リモートポート転送（受信ポート転送）は、指定されたリモートポートからの受信データを、安全なチャネルを介して、指定されたローカルポートに送信します。

漢字

暗号

暗号とは暗号化アルゴリズムのことです。選択した暗号によって、Secure Shell 接続の確立完了後に送信されるデータの暗号化に使用される数学アルゴリズムが決定されます。

暗号化

暗号化とは、暗号すなわち秘密のコードを使用してデータを加工し、許可されたユーザ以外には解読できないようにすることです。暗号化されていないデータに比べ、暗号化されたデータははるかに安全です。

公開鍵と秘密鍵

公開鍵と秘密鍵は、データの暗号化または解読に使用される暗号鍵のペアです。公開鍵で暗号化されたデータは、秘密鍵を使用した場合のみ解読できます。また、秘密鍵で暗号化されたデータは、公開鍵を使用した場合のみ解読できます。

正規表現

正規表現は、1 つ以上の一致する文字列を記述する文字列です。よく *regex* と省略されます。正規表現内では、一部の文字は事前に定義された意味を持ち、何が一致と見なされるかを決定します。たとえば、正規表現「t.*t」は、文字 *t* で始まり、かつ終わるすべての単語に一致します。一方、正規表現「text」はそれ自体のみに一致します。

電子証明書

PKI（公開鍵インフラストラクチャ）には不可欠な部分です。電子証明書（別名、X.509 証明書）は認証局（CA）によって発行されるもので、証明書内の情報の有効性を保証します。証明書は、その証明書所有者の識別情報、証明書所有者の公開鍵のコピー（メッセージやデジタル署名の暗号化および復号化に使用される）、およびデジタル署名（証明書の内容に基づいて、CA が生成したもの）が含まれています。受信者はデジタル署名を使用して、証明書が改ざんされておらず、信頼できることを確認できます。

認証

通信相手の身元を確実に確認する処理。身元の確認は、パスワードなどの既知の情報、または秘密鍵やトークンなど所有しているもの、指紋などの固有の情報を使用して行います。

索引

あ

た

は

や

ら

漢字