NEC
NECソリューションイノベータ

InfoCage FileShell クライアント API
利用ガイド

InfoCage

InfoCage FileShell
Version 6.3
クライアントAPI 利用ガイド
（0630A01）

はじめに

このたびは、NEC ソリューションイノベータ株式会社の InfoCage FileShell をお買い求めいただき誠にありがとうございます。

InfoCage FileShell は、電子ファイル自身にセキュリティ情報を持たせた暗号化をおこなうことで、

利用者の操作性を損なうことなく重要な情報を永続的に保護する機密情報保護ソフトウェアです。

ご使用になる前に本書をよくお読みになり、製品の取り扱いを十分にご理解ください。

■ 商標について

- Microsoft および Windows は米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。
- InfoCage は NEC ソリューションイノベータ株式会社の登録商標です。
- その他、本書に記載されている会社名、商品名は各社の登録商標または商標です。

■ 免責事項

本書および本システムは、ライセンス契約に基づいて使用することができます。
ライセンス契約で明示的に定められていないかぎり、NEC ソリューションイノベータ株式会社は製品、およびその関連文書について、明示的にも暗黙的にも、商品性に関する保証、特定目的への適合性に関する保証、取り扱い、使用、または取引行為に伴う保証について一切の責任を負いません。

本書中のサンプル画面で使用している名称は、すべて架空のものです。実在する品名、団体名、個人名とは一切関係ありません。

本書について

本書は本製品を正しく運用し、効果的に活用するための手引きです。運用を開始する前や運用中に、機能・操作を確認するためにご利用ください。

本書は、InfoCage FileShell クライアント API を使ったアプリケーションの開発者、および、InfoCage FileShell クライアント API を使ったシステムの運用者を対象としています。

ご注意： 本書の一部、または全部を流用・複写することはできません。

本書中の記号について

本書中では、以下の記号を使用しています。これらの記号の意味を正しくご理解になり、
本書をお読みください。

記　号	説　　明
Notice	システムの取り扱いで守らなければならない事柄や特に注意すべき点、確認すべき点を説明します。
参照	関連する内容が記載されているページを紹介しています。
Operation	操作手順を示します。

参考資料について

本書中では、参考資料として以下のガイドを参照するように説明しています。

項　目	ガ　イ　ド　名	番　号
インストールガイド	InfoCage FileShell インストールガイド	0630Snn
FileShell SDK 利用ガイド	InfoCage FileShell SDK 利用ガイド	0630Dnn

＊ 末尾の「nn」には、「01」、「02」などの数字(版数)が入ります。
版数は、プログラムやマニュアルに変更があった場合に更新されます。

用語の定義

本書で使用されている用語については、管理者ガイドの「用語の定義」をご参照ください。

目 次

第 1 章　FileShell クライアント API について 1

1.1　特長 1

1.2　機能 1

1.3　動作環境について 3

第 2 章　注意事項 4

2.1　運用上の注意事項 4

第 3 章　アプリケーションの開発 5

3.1　開発環境への展開 5

3.2　使用方法 5

3.3　API リファレンス 6

3.3.1　ファイル分類/保護（Office IRM/FileShell、Microsoft 互換/FileShell(ラベル)、NFP 形式）－ IcfcProtect() 7

3.3.2　ファイル保護（マルチデバイス形式）－ IcfcProtectMultiDeviceFormat() 9

3.3.3　ファイル分類/保護解除（Office IRM/FileShell、Microsoft 互換/FileShell(ラベル)、NFP 形式）－ IcfcUnprotect() 10

3.3.4　ファイル保護解除（マルチデバイス形式）－ IcfcUnprotectMultiDeviceFormat() 11

3.3.5　メモリ解放 － IcfcFreeMemory() 12

3.3.6　分類/保護状態確認 － IcfcIsProtected() 13

3.3.7　分類状態確認 － IcfcIsLabeled() 14

3.3.8　保護形式確認 － IcfcIsMultiDeviceFormatFile() 15

3.3.9　コンテンツ ID 取得 － IcfcGetContentId() 16

3.3.10　権限取得 － IcfcGetRights() 17

3.3.11　権利ポリシー作成 － IcfcCreateRightsPolicyFile() 18

3.3.12　ファイルフォーマット取得－ IcfcGetFileFormat() 19

3.4　エラーコード 20

3.5　サンプルコード 23

第 4 章　FileShell クライアント API の展開 27

4.1　FileShell クライアント API の展開 27

4.2　権利ポリシーテンプレートの配置と保護時のパス指定 28

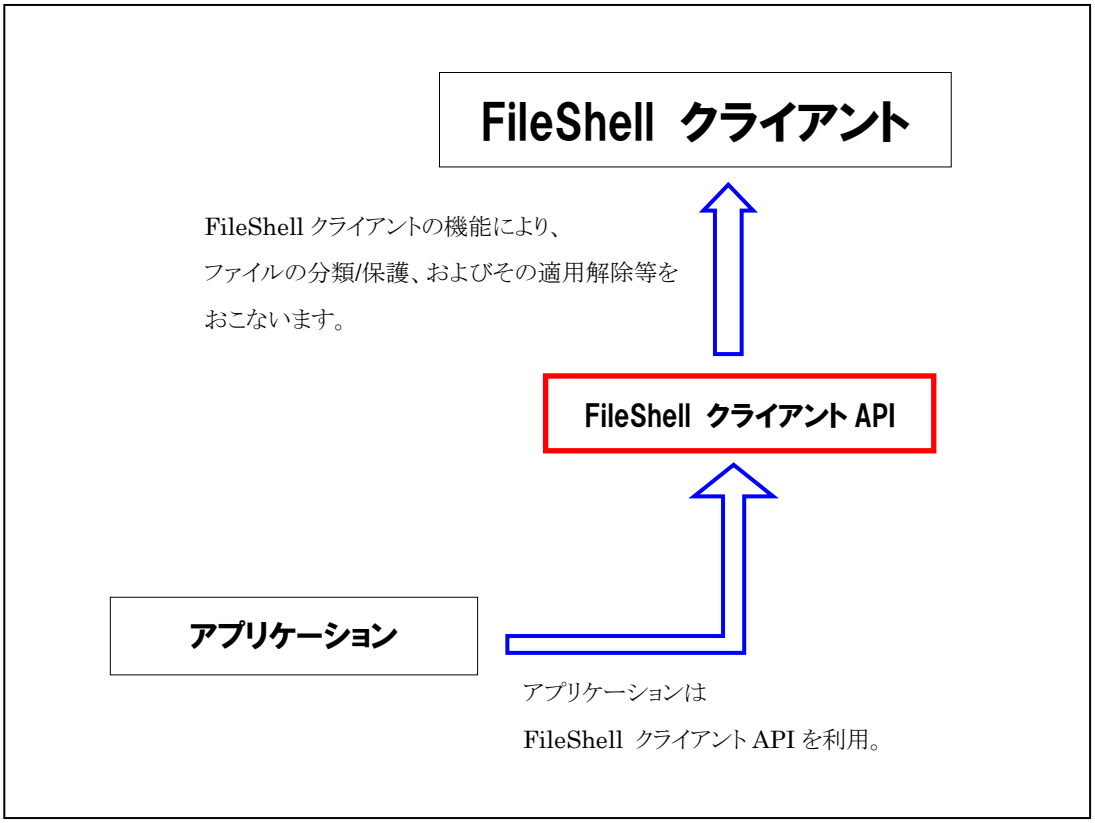
4.2.1　Office IRM/FileShell 形式またはマルチデバイス形式で保護をおこなう場合 28

4.2.2　NFP 形式で保護をおこなう場合 28

第1章 FileShell クライアント API について

1.1 特長

FileShell クライアント API は FileShell クライアントが動作するクライアントマシン上で、ファイルの分類/保護、およびその適用解除等をおこなうための API を提供します。



1.2 機能

FileShell クライアント API は、以下の機能を提供します。各形式ごとに使用する機能は以下のとおりです。

OfficeIRM/FileShell 形式、Microsoft 互換/FileShell(ラベル)形式

機能名	概要	関数インタフェース名	参照
ファイル分類/保護	指定されたファイルを分類/保護します。	IcfcProtect()	3.3.1
ファイル分類/保護解除	指定された分類/保護ファイルの適用を解除します。	IcfcUnprotect()	3.3.3
分類/保護状態確認	指定されたファイルが分類/保護済みかどうかの確認をおこないます。	IcfcIsProtected()	3.3.6
分類状態確認	指定されたファイルが分類(ラベル付与のみ)済みかどうかの確認をおこないます。	IcfcIsLabeled()	3.3.7

コンテンツ ID 取得	指定された分類/保護ファイルのコンテンツ ID※1を取得します。	IcfcGetContentId()	3.3.9
権限取得	指定された分類/保護ファイルに対する、利用者の権限を取得します。	IcfcGetRights()	3.3.10
権利ポリシー作成	ローカル権利ポリシーを作成します。	IcfcCreateRightsPolicyFile()	3.3.11
ファイルフォーマット取得	分類/保護されたファイルの出力形式を取得します。	IcfcGetFileFormat()	3.3.12

マルチデバイス形式

機能名	概要	関数インタフェース名	参照
ファイル保護	指定されたファイルをマルチデバイス形式で保護します。	IcfcProtectMultiDeviceFormat()	3.3.2
ファイル保護解除	指定されたマルチデバイス形式の保護ファイルの保護を解除します。	IcfcUnprotectMultiDeviceFormat()	3.3.4
メモリ解放	マルチデバイス形式でファイルを保護または保護解除する際に確保したメモリを解放します。	IcfcFreeMemory()	3.3.5
保護状態確認	指定されたファイルが保護済みかどうかの確認をおこないます。	IcfcIsProtected()	3.3.6
保護形式確認	指定されたファイルがマルチデバイス形式で保護済みかどうかの確認をおこないます。	IcfcIsMultiDeviceFormatFile()	3.3.7
コンテンツ ID 取得	指定された保護ファイルのコンテンツ ID※1を取得します。	IcfcGetContentId()	3.3.9
権限取得	指定された保護ファイルに対する、利用者の権限を取得します。	IcfcGetRights()	3.3.10
ファイルフォーマット取得	保護されたファイルの保護形式を取得します。	IcfcGetFileFormat()	3.3.12

NFP 形式

機能名	概要	関数インタフェース名	参照
ファイル保護	指定されたファイルを保護します。	IcfcProtect()	3.3.1
ファイル保護解除	指定された保護ファイルの保護を解除します。	IcfcUnprotect()	3.3.3
保護状態確認	指定されたファイルが保護済みかどうかの確認をおこないます。	IcfcIsProtected()	3.3.6
コンテンツ ID 取得	指定された保護ファイルのコンテンツ ID※1を取得します。	IcfcGetContentId()	3.3.9
権限取得	指定された保護ファイルに対する、利用者の権限を取得します。	IcfcGetRights()	3.3.10
ファイルフォーマット取得	保護されたファイルの保護形式を取得します。	IcfcGetFileFormat()	3.3.12

(※1) コンテンツ ID ： ファイルを保護した際に一意に設定される識別コード(GUID)です。

1.3 動作環境について

FileShell クライアント API の動作環境は以下のとおりです。以下に記載がない事項については、『インストールガイド』の「動作環境について」内の「FileShell クライアント」をご参照ください。

■ FileShell クライアント API 動作環境

必要環境	
FileShell 製品、その他	FileShell クライアント API を動作させるためには、FileShell クライアントの動作環境が必要です。

■ 連携可能な InfoCage FileShell 製品バージョン

対応バージョン	
FileShell クライアント	V6.0、V6.1、V6.2、V6.3

＊ V6.0 未満の FileShell クライアントとの連携には使用できません。V6.0 未満のバージョンの FileShell クライアントとの連携には、連携する FileShell クライアントと同じバージョンの FileShell クライアント API をご使用ください。

第2章 注意事項

2.1 運用上の注意事項

＊ FileShell クライアントAPI では、FileShell クライアントの利用アカウントを使ってファイルの分類/保護、およびその適用解除等の処理をおこなっています。このため、FileShell クライアント API を利用するアプリケーションの実行時には、以下の注意事項があります。

・FileShell クライアントAPIを利用するアプリケーションは、Windows のログオンアカウントで実行する必要があります。

・FileShell クライアント API を利用時には、その実行アカウントで Windows にログオンした状態にしておく必要があります。

・FileShell クライアント API を利用してファイルの分類/保護の適用解除をおこなう場合、FileShell クライアントの利用アカウントにはその分類/保護ファイルに対するフルコントロール権限が必要となります。

・ドメインログオンしていない場合、あるいは、インターネットオプションのセキュリティの設定で自動認証の設定になっていない場合には、認証ダイアログが表示される場合があります。もし利用者が正常に認証をおこなわなかった場合、FileShell クライアント API は処理に失敗し、エラーとなります。

＊ FileShell クライアント API の入力ファイル（分類/保護、およびその適用解除しようとしているファイル等）は FileShell クライアントの動作仕様上、API の処理内で排他ロックされる場合があります。そのため、FileShell クライアントで処理中のファイルを API の入力ファイルとした場合、あるいは、FileShell クライアント API を同時に複数実行して同じファイルを処理しようとした場合には、エラーとなる場合があります。

＊ 64bitOS 上で 32bit アプリケーションを動作させる場合は、32bit アプリケーション用のモジュールをご利用いただく必要があります。

＊ RMS サーバーで発行された権利ポリシーテンプレートを使用する場合は、
%localappdata%¥Microsoft¥MSIPC¥UnmanagedTemplates にも同じファイルを配置しておく必要があります。
ローカル権利ポリシーテンプレートを使用する場合は、任意のパスに配置してください。
ただし、上記フォルダー(UnmanagedTemplates フォルダー)には配置しないでください。

＊ FileShell クライアント API で%localappdata%¥Microsoft¥MSIPC¥UnmanagedTemplates に権利ポリシーテンプレートを配置し利用する場合、FileShell ポリシーで「クライアントで RMS サーバーから権利ポリシーを取得できるようにする」設定は有効にできません。（有効にした場合、FileShell が正常に動作しない場合があります。）

＊ ログオン直後などでクライアントのプロセスが起動していないときにクライアント API の呼び出しをおこなうと、以下のエラーが発生する場合があります。

0x800704db(HRESULT_FROM_WIN32(ERROR_SERVICE_NOT_FOUND)指定されたサービスはありません。）

その場合は、しばらくたってからもう一度クライアント API の呼び出しをおこなってください。

＊ マルチデバイス形式保護 API を使用して Office 形式ファイルを保護した場合、Office IRM 形式として保護されます。

＊ パスワードで保護された Office 2007 形式ファイルは、マルチデバイス形式保護 API を使用して保護することはできません。

第3章 アプリケーションの開発

本章では、FileShell クライアント API を使ったアプリケーションの開発方法を紹介します。

3.1 開発環境への展開

アプリケーションの開発環境に、FileShell クライアント API のヘッダファイルとライブラリファイルを展開します。

Operation

1. FileShell クライアント API の Include フォルダー、Lib フォルダーにあるヘッダファイルとライブラリファイルをアプリケーションの開発環境にコピーします。

   参照 FileShell クライアント API に含まれるフォルダー構成については「4.1 FileShell クライアント API の展開」を参照してください。

3.2 使用方法

FileShell クライアント API を使用するには、以下の2種類の方法があります。

1. 利用する C/C++アプリケーションよりヘッダファイル(IcfcProtector.h)をインクルードしてコンパイルし、インポートライブラリ(IcfcProtector.lib)をリンクして使用する。
2. 利用するアプリケーションより IcfcProtector.dll をロードライブラリで動的にロードして使用する。

Notice

- ヘッダファイル、ライブラリは C/C++の形式です。
- IcfcProtector.dll は FileShell クライアントが動作するマシンの任意のフォルダーから実行可能です。

3.3 API リファレンス

FileShell クライアント API のインタフェースは、それぞれ単独で呼び出すことが可能です。

3.3.1 ファイル分類/保護（Office IRM/FileShell、Microsoft 互換/FileShell(ラベル)、NFP 形式）－ IcfcProtect()

《概要》

Office IRM/FileShell 形式、Microsoft 互換/FileShell(ラベル)形式、または NFP 形式でファイルの分類/保護をおこないます。ファイルに付与する権限情報は、権利ポリシーテンプレート、またはラベル ID で指定します。
ファイルの分類/保護形式は、pszRptPath に指定する権利ポリシーテンプレート、またはラベル ID によって自動的に判断されます。

参照 権利ポリシーテンプレートの作成方法は、『FileShell SDK 利用ガイド』の「権利ポリシーテンプレートの準備」の項目を参照してください。
ラベル ID の取得方法は、『FileShell SDK 利用ガイド』の「ラベル ID の取得」の項目を参照してください。

《構文》

```
HRESULT WINAPI IcfcProtect(
    LPCWSTR     pszFileSrc,     // [in] 分類/保護するファイルパス
    LPCWSTR     pszFileDst,     // [in] 分類/保護後の出力ファイルパス
    LPCWSTR     pszRptPath      // [in] 付与する権利ポリシーテンプレートファイルパス、
                                        または分類に使用するラベルのラベル ID (Guid)
);
```

《引数》

pszFileSrc [in]　分類/保護するファイル名(ドライブ名を含む絶対パス)を指定します。

pszFileDst [in]　分類/保護後の出力ファイル名(ドライブ文字を含む絶対パス)を指定します。ここで指定された出力ファイル名で保護ファイルが出力されます。分類/保護するファイル名と、分類/保護後の出力ファイル名を同一にすることはできません。

pszRptPath [in]　Office IRM/FileShell 形式で保護する場合
ドライブ文字を含む絶対パスで、任意のフォルダーに格納されているファイルの権限情報が記述された権利ポリシーテンプレートファイルを指定します。

Microsoft 互換/FileShell(ラベル)形式で分類/保護する場合
分類に使用するラベル ID(Guid) を、"{ }"付きで指定)します。

指定例:
IcfcProtect(pszSrc, pszDst, L"{deb36b0e-9fac-41a5-8c9e-eca546ad3459}");

＊ 適用時にユーザーによるアクセス許可の割り当てがおこなえるラベルは指定できません。
＊ ラベル ID の GUID は 取得時の文字列のまま指定してください。(大文字などにするとエラーとなります)

NFP 形式で保護する場合
ドライブ文字を含む絶対パスで、以下のいずれかのフォルダーに格納されている権利ポリシーテンプレートファイルを指定します。

(環境変数で記載の個所は、ご使用の環境の実際のパスに置き換えて指定してください)

%AppData%¥NEC¥InfoCageFileShell¥Policy

%APPDATA%¥NEC¥InfoCageFileShell¥EncPolicy
%AllUsersProfile%¥NEC¥InfoCageFileShell¥Policy

指定するNFP権利ポリシーテンプレートの確認方法については「4.2 権利ポリシーテンプレートの配置と保護時のパス指定」の「4.2.2 NFP 形式で保護をおこなう場合」を参照してください。

《戻り値》

関数が成功した場合、S_OK が返却されます。失敗した場合、Windows 標準、RMS または FileShell のエラー値が返却されます。

3.3.2　ファイル保護（マルチデバイス形式）－ IcfcProtectMultiDeviceFormat()

《概要》

マルチデバイス形式でファイルの保護をおこないます。ファイルに付与する権限情報は、権利ポリシーテンプレートで指定します。

参照　権利ポリシーテンプレートの作成方法は『FileShell SDK 利用ガイド』の「権利ポリシーテンプレートの準備」の項目を参照してください。

《構文》

```
HRESULT WINAPI IcfcProtectMultiDeviceFormat(
    LPCWSTR     pszFileSrc,          // [in] 保護するファイルパス
    LPCWSTR     pszOutputDir,        // [in] 保護されたファイルを出力するフォルダーパス
    LPCWSTR     *ppszOutputFilePath, // [out] 保護後の出力ファイルパス
    LPCWSTR     pszRptPath           // [in] 付与する権利ポリシーテンプレートファイルパス
);
```

＊ ppszOutputFilePath 利用後は、IcfcFreeMemory で解放する必要があります。

参照　ppszOutputFilePath については、「3.3.5 メモリ解放 － IcfcFreeMemory()」を参照してください。

＊ RMS サーバーで発行された権利ポリシーテンプレートを使用する場合は、%allusersprofile%¥Microsoft¥MSIPC¥Server¥UnmanagedTemplates¥<SID>にも同じファイルを配置しておく必要があります。ローカル権利ポリシーテンプレートを使用する場合は、上記フォルダーには配置しないでください。

《引数》

pszFileSrc [in]	保護するファイル名(ドライブ名を含む絶対パス)を指定します。
pszOutputDir [in]	保護されたファイルを出力するフォルダー(ドライブ名を含む絶対パス)を指定します。
ppszOutputFilePath [out]	保護後のファイルパスが返却されます。
pszRptPath [in]	ファイルの権限情報が記述された権利ポリシーテンプレートファイル名(ドライブ文字を含む絶対パス)を指定します。

《戻り値》

関数が成功した場合、S_OK が返却されます。失敗した場合、Windows 標準、RMS または FileShell のエラー値が返却されます。

3.3.3 ファイル分類/保護解除（Office IRM/FileShell、Microsoft 互換/FileShell(ラベル)、NFP 形式）－ IcfcUnprotect()

《概要》

Office IRM/FileShell 形式、Microsoft 互換/FileShell(ラベル)形式、または NFP 形式で分類/保護されているファイルへの適用を解除します。

《構文》

```
HRESULT WINAPI IcfcUnprotect(
    LPCWSTR     pszFileSrc,     // [in] 分類/保護の適用を解除するファイルパス
    LPCWSTR     pszFileDst      // [in] 分類/保護の適用解除後の出力ファイルパス
);
```

《引数》

pszFileSrc [in]　分類/保護の適用を解除するファイル名(ドライブ名を含む絶対パス)を指定します。

pszFileDst [in]　分類/保護の適用解除後の出力ファイル名(ドライブ文字を含む絶対パス)を指定します。ここで指定された出力ファイル名で分類/保護の適用解除後のファイルが出力されます。分類/保護の適用を解除するファイル名と分類/保護の適用解除後の出力ファイル名を同一にすることはできません。

《戻り値》

関数が成功した場合、S_OK が返却されます。失敗した場合、Windows 標準、RMS または FileShell のエラー値が返却されます。

3.3.4　ファイル保護解除（マルチデバイス形式）－ IcfcUnprotectMultiDeviceFormat()

《概要》

マルチデバイス形式で保護されているファイルの保護を解除します。

《構文》

```
HRESULT WINAPI IcfcUnprotectMultiDeviceFormat(
    LPCWSTR        pszFileSrc,          // [in] 保護解除するファイルパス
    LPCWSTR        pszOutputDir,        // [in] 保護解除されたファイルを出力するフォルダーパス
    LPCWSTR        *ppszOutputFilePath, // [out] 保護解除後の出力ファイルパス
);
```

＊　ppszOutputFilePath 利用後は、IcfcFreeMemory で解放する必要があります。

参照　ppszOutputFilePath については、「3.3.5 メモリ解放 － IcfcFreeMemory()」を参照してください。

《引数》

pszFileSrc [in]	保護解除するファイル名(ドライブ名を含む絶対パス)を指定します。
pszOutputDir [in]	保護解除されたファイルを出力するフォルダー(ドライブ名を含む絶対パス)を指定します。
ppszOutputFilePath [out]	保護後のファイルパスが返却されます。

《戻り値》

関数が成功した場合、S_OK が返却されます。失敗した場合、Windows 標準、RMS または FileShell のエラー値が返却されます。

3.3.5　メモリ解放 ― IcfcFreeMemory()

《概要》

マルチデバイス形式でファイルを保護または保護解除する際、ppszOutputFilePath で確保したメモリを解放します。

＊ OfficeIRM/FileShell 形式または NFP 形式で保護または保護解除する場合は実行不要です。

《構文》

```
void WINAPI IcfcFreeMemory(
    LPCWSTR          pszOutputFilePath,    // [in] 解放するメモリのポインタ

);
```

《引数》

pszOutputFilePath[in]　　IcfcProtectMultiDeviceFormat または IcfcUnprotectMultiDeviceFormat にて取得したファイルパスのメモリを解放します。

＊ ppszOutputFilePath を指定してください。

《戻り値》

なし。

3.3.6 分類/保護状態確認 ― IcfcIsProtected()

《概要》

ファイルが OfficeIRM/FileShell 形式、Microsoft 互換/FileShell(ラベル)形式、マルチデバイス形式または NFP 形式で分類/保護済みかどうかの確認処理をおこないます。

《構文》

```
HRESULT WINAPI IcfcIsProtected(
    LPCWSTR     pszFile,        // [in] 確認するファイルパス
    BOOL        *pbRet          // [out] 分類/保護されているかどうか
);
```

《引数》

pszFile [in]	分類/保護状態を確認するファイル名(ドライブ文字を含む絶対パス)を指定します。
pbRet [out]	分類/保護の有無についてチェックした結果が格納されます。 分類/保護されている場合は TRUE、分類/保護されていない場合は FALSE が格納されます。

＊ Microsoft 互換形式で分類(ラベル付与のみ)されているファイルは FALSE となります。

《戻り値》

関数が成功した場合、S_OK が返却されます。失敗した場合、Windows 標準、RMS または FileShell のエラー値が返却されます。

Notice

- 本 API では、分類/保護されているファイルとして必要なデータが確認できない場合、分類/保護されていないファイルと判定します。そのため、必要なデータにアクセスできない場合やファイルが存在しない場合でも、戻り値は成功となり、pbRet に FALSE が返却されます。

3.3.7 分類状態確認 ― IcfcIsLabeled()

《概要》

ファイルが分類(ラベル付与のみ)されているかどうかの確認処理をおこないます。

《構文》

```
HRESULT WINAPI IcfcIsLabeled(
LPCWSTR        pszFile,        // [in] 確認するファイルパス
BOOL           *pbRet          // [out] 分類(ラベル付与)されているかどうか
);
```

《引数》

pszFile [in]　分類(ラベル付与)を確認するファイル名(ドライブ文字を含む絶対パス)を指定します。

pbRet [out]　分類(ラベル付与)についてチェックした結果が格納されます。
分類(ラベル付与)されている場合は TRUE、分類(ラベル付与)されていない場合は FALSE が格納されます。

＊ Microsoft 互換形式で分類(ラベル付与のみ)されているファイルのみ TRUE となります。

＊ 分類されていても、Microsoft 互換/FileShell(ラベル)形式で保護されているファイルは FALSE となります。

《戻り値》

関数が成功した場合、S_OK が返却されます。失敗した場合、Windows 標準、RMS または FileShell のエラー値が返却されます。

3.3.8 保護形式確認 ― IcfcIsMultiDeviceFormatFile()

《概要》

ファイルがマルチデバイス形式で保護済みかどうかの確認処理をおこないます。

《構文》

```
HRESULT WINAPI IcfcIsMultiDeviceFormatFile(
    LPCWSTR     pszFile,        // [in] 確認するファイルパス
    BOOL        *pbRet          // [out] 保護されているかどうか
);
```

《引数》

pszFile [in]　保護形式を確認するファイル名(ドライブ文字を含む絶対パス)を指定します。

pbRet [out]　保護形式についてチェックした結果が格納されます。
マルチデバイス形式の場合は TRUE、保護されていないまたはマルチデバイス形式でない場合は FALSE が格納されます。

＊ 以下の拡張子については、Microsoft 互換/FileShell(ラベル)形式で保護されたファイルでも、マルチデバイス形式として認識されます。
pfile,ptxt,pxml,pjpg,pjpeg,ppng,ptif,ptiff,pbmp,pgif,pjpe,pjfif,pjt,pjif,pjfi

《戻り値》

関数が成功した場合、S_OK が返却されます。失敗した場合、Windows 標準、RMS または FileShell のエラー値が返却されます。

3.3.9 コンテンツ ID 取得 ― IcfcGetContentId()

《概要》

分類/保護ファイルのコンテンツ ID 取得処理をおこないます。

《構文》

```
HRESULT WINAPI IcfcGetContentId(
    LPCWSTR     pszFile,        // [in] 取得するファイルパス
    GUID        *pContentId     // [out] コンテンツ ID
);
```

《引数》

pszFile [in]	コンテンツ ID を取得する分類/保護ファイル名(ドライブ名を含む絶対パス)を指定します。
pContentId [out]	分類/保護ファイルのコンテンツ ID が格納されます。

＊ Microsoft 互換形式として保護された PDF ファイルはコンテンツ ID を取得できません。

《戻り値》

関数が成功した場合、S_OK が返却されます。失敗した場合、Windows 標準、RMS または FileShell のエラー値が返却されます。

3.3.10 権限取得 ― IcfcGetRights()

《概要》

分類/保護ファイルに対する、利用者の権限を取得します。

《構文》

```
HRESULT WINAPI IcfcGetRights(
    LPCWSTR     pszFile,        // [in] 権限取得するファイルパス
    DWORD       *pdwRights      // [out] 権限
);
```

《引数》

pszFile [in]　　権限を取得する分類/保護ファイル名(ドライブ名を含む絶対パス)を指定します。

pdwRights [out]　　分類/保護ファイルに対する、利用者の権限が格納されます。
権限は以下ビットの論理和になります。

権限	
フルコントロール	0x00000001
表示	0x00000002
編集	0x00000004
保存	0x00000008
エクスポート	0x00000010
印刷	0x00000020
転送	0x00000040
返信	0x00000080
全員へ返信	0x00000100
抽出	0x00000200
マクロの許可	0x00000400
権利の表示	0x00000800
権利の編集	0x00001000

《戻り値》

関数が成功した場合、S_OK が返却されます。失敗した場合、Windows 標準、RMS または FileShell のエラー値が返却されます。

3.3.11 権利ポリシー作成 ― IcfcCreateRightsPolicyFile()

《概要》

FileShell 権利ポリシー構造体に指定した内容に従って、ローカル権利ポリシーファイル（xml）を生成します。

《構文》

```
HRESULT WINAPI IcfcCreateRightsPolicyFile(
    ICF_RIGHTS_POLICY        *pRightsPolicy,      // [in] FileShell 権利ポリシー構造体
    LPCWSTR                  pszFilePath          // [in] 権利ポリシーファイルの出力ファイルパス
);
```

《引数》

pRightsPolicy [in]　　FileShell 権利ポリシー構造体（ICF_RIGHTS_POLICY）を指定します。あらかじめ、指定したい権限に従って値を設定しておく必要があります。

参照　FileShell 権利ポリシー構造体の指定方法は『FileShell SDK 利用ガイド』の「FileShell 権利ポリシー構造体 ― ICF_RIGHTS_POLICY」を参照してください。

pszFilePath [in]　　ローカル権利ポリシーファイルの出力ファイル名(ドライブ文字を含む絶対パス)を指定します。ここで指定された出力ファイル名でローカル権利ポリシーファイル（xml）が出力されます。

《戻り値》

関数が成功した場合、S_OK が返却されます。失敗した場合、Windows 標準、RMS または FileShell のエラー値が返却されます。

3.3.12 ファイルフォーマット取得－ IcfcGetFileFormat()

《概要》

分類/保護されたファイルの出力形式を取得します。

《構文》

```
HRESULT IcfcGetFileFormat(
    LPCWSTR     pszFile,        // [in] 分類/保護されたファイル名
    DWORD       *pdwFormat      // [out] ファイルフォーマット
    );
```

《引数》

pszFile [in]　　保護されたファイル名（ドライブ名を含む絶対パス）を指定します。

**pdwFormat* [out]　　*pszFile* で指定したファイルの保護形式が格納されます。格納される値と保護形式の関係は以下のとおりです。

0: 保護されていない

1: Office IRM/FileShell 形式

2: マルチデバイス形式

＊ 以下の拡張子については、Microsoft 互換/FileShell(ラベル)形式で保護されたファイルでも、マルチデバイス形式として認識されます。
pfile,ptxt,pxml,pjpg,pjpeg,ppng,ptif,ptiff,pbmp,pgif,pjpe,pjfif,pjt,pjif,pjfi

4: NFP 形式

16: Microsoft 互換/FileShell 形式（ラベル付与による分類と保護）

32: Microsoft 互換形式（ラベル付与による分類のみ）

《戻り値》

関数が成功した場合、S_OK が返却されます。失敗した場合、Windows、RMS または FileShell のエラー値が返却されます。

3.4 エラーコード

FileShell クライアントAPIが提供する関数は、エラーコードとして、winerr.hで定義される通常のエラーコード、もしくは下記 URL で定義される AD RMS 独自のエラーコードを返却します。

AD RMS Function Error Codes (2023/9/1 時点)
http://msdn.microsoft.com/en-us/library/bb204613

その他、FileShell 独自のエラーコードとして以下が返却される場合があります。

エラーコード	原因	対処策
0x00040302	既にファイルが保護されているファイルを保護しようとした。	原因欄に記載の内容のとおりです。
0x00040303	既にファイルが保護解除されているファイルを保護解除しようとした。	同上
0x00040305	サイズが 0 バイトのファイルを保護しようとした。	サイズが 0 バイトのファイルは保護できません。
0x80040500	指定された権限で既にファイルが保護されている。	原因欄に記載の内容のとおりです。
0x0004030B	保護または保護解除後のファイル名と同じ名前のファイルが存在します。	マルチデバイス形式での保護・保護解除後、ファイルの拡張子が変更される場合があります。同じ名前のファイルが存在するか確認後、再度保護・保護解除してください。
0x0004030D	権利ポリシーテンプレート情報が見つからない。	保護対象フォルダーの設定に問題がある可能性があります。詳しくは管理者にお問い合わせください。
0x0004030E	原因不明のエラーが発生しました。破損したファイルである可能性があります。	事象ごとに個別に調査する必要があります。詳しくは管理者にお問い合わせください。
0x0004030F	ネットワークに接続されていないか、サーバーが見つからない。	ネットワークの設定を確認してください。 ネットワークドライブをアンマウントしていないか、最近接続を解除した(未認証状態)ネットワークがないか、確認してください。
0x80040800	Excel 4.0 マクロシート入りファイルを保護しない設定を実施しているにもかかわらず、同ファイルを保護しようとした。	保護したい場合は、保護しない設定を解除してください。
0x80040801	Excel 5.0 モジュールシート入りファイル保護しない設定を実施しているにもかかわらず、同ファイルを保護しようとした。	同上
0x80040802	アドインファイルを保護しない設定を実施しているにもかかわらず、同ファイルを保護しようとした。	同上
0x80040803	ファイル構造と拡張子が一致しない。	拡張子が変更されていないか、ご確認ください。
0x80040900	RMS Client 2.1 がインストールされていない。	RMS Client 2.1 をインストールしてください。

0x80040901	認証ダイアログでキャンセルした。	認証を実施してください。
0x80040951	時刻の巻き戻しが検知された。	設定されている時刻が正しいか、ご確認ください。
0x80040952	未変換 NFP 権利ポリシーのインポートをキャンセルした。	NFP 権利ポリシーをインポートしてください。
0x80040953	NFP の緊急保護解除機能で使用する公開鍵の取得に失敗した。	NFP の緊急保護解除機能をオフと設定するか、緊急保護解除機能用の公開鍵を設定してください。
0x80040954	FileShell ポリシーの受信が必要(有効期間が切れた)。	FileShell ポリシーの受信を実施してください。
0x80042000	マルチデバイス形式ファイルでない。	マルチデバイス形式ファイルを指定してください。
0x80042001	マルチデバイス形式で保護または保護解除するとき、出力ファイル名が入力ファイル名と同一になる。	ファイル名をご確認ください。
0x80044000	無効な入力エラー。	AIP の引数を見直してください。
0x80044001	バッファーのメモリ不足。	AIP の引数を見直してください。
0x80044002	ファイル IO エラー。	AIP に与えたファイルの状態を確認してください。
0x80044003	ネットワーク エラー。	ネットワークの状態を確認してください。
0x80044004	内部エラー。	AIP の引数や Microsoft Purview コンプライアンス ポータルにてラベルの状態を見直してください。
0x80044005	ファイルに対するアクションが完了できない。	Microsoft Purview コンプライアンス ポータルにてラベルの状態を見直してください。
0x80044007	現在のラベルは特権操作（管理者の操作と同等）として割り当てられている。	Microsoft Purview コンプライアンス ポータルにてラベルの設定を見直してください。
0x80044008	ユーザーがコンテンツにアクセスできない。	Microsoft Purview コンプライアンス ポータルにてラベルの設定を見直してください。
0x80044009	同意が得られなかった。	Microsoft Purview コンプライアンス ポータルにてラベルの設定を見直してください。
0x80044010	コンテンツにアクセスできない。	ネットワークの状態を確認してください。
0x80044011	認証トークンがない。	Microsoft Purview コンプライアンス ポータルにてラベルの設定を見直してください。
0x80044012	サービスが無効になっている。	Microsoft Purview コンプライアンス ポータルにてラベルの状態を見直してください。
0x80044013	プロキシ認証エラー。	ネットワークの状態を確認してください。
0x80044014	テナントポリシーが分類/ラベルに対して構成されていない。	Microsoft Purview コンプライアンス ポータルにてラベルの設定を見直してください。

0x80044015	操作が取り消された。	再試行してください。
0x80044016	アドホック保護を設定する必要がある。	別のラベルを指定してください。
0x80044017	呼び出し元が非推奨の API を呼び出した。	再インストールしてください。
0x80044018	テンプレート ID が RMS サービスによって認識されない。	Microsoft Purview コンプライアンスポータルにてラベルの設定を見直してください。
0x80044019	ラベル ID が認識されていない。	Microsoft Purview コンプライアンスポータルにてラベルの設定を見直してください。
0x80044020	ラベルが無効または非アクティブ。	Microsoft Purview コンプライアンスポータルにてラベルの設定を見直してください。
0x80044021	ダブルキー機能が有効になっていない。	Microsoft Purview コンプライアンスポータルにてラベルの設定を見直してください。
0x80044022	ライセンスが登録されていない。	Microsoft Purview コンプライアンスポータルにてラベルの設定を見直してください。
0x80044501	アドホックのラベルは使用できない。	別のラベルを指定してください。
0x80044502	既に保護されている。	保護を解除してからやり直してください。

3.5 サンプルコード

FileShell クライアント API を利用したサンプルコードを掲載します。

・サンプル1

以下のサンプルでは、FileShell 形式に関する 権利ポリシー作成（IcfcCreateRightsPolicyFile）、ファイルの保護（IcfcProtect）、保護状態確認（IcfcIsProtected）、コンテンツ ID 取得（IcfcGetContentId）、権限の取得（IcfcGetRights）、保護解除（IcfcUnprotect）という一連の処理をおこなっています。

```
#include "stdafx.h"
#include <windows.h>
#include "icfcprotector.h"

#pragma comment(lib, "icfcprotector.lib")

int _tmain(int argc, _TCHAR* argv[])
{
	BOOL bRet = FALSE;
	GUID guid;
	HRESULT hr;
	DWORD dwRights = 0;

	ICF_RIGHTS_POLICY sRightsPolicy = {0};
	ICF_NAME_AND_DESC sNameAndDesc = {0};
	ICF_USER_AND_RIGHT sUserAndRight = {0};
	BOOL bOwnerFullControl = TRUE;
	UINT uiEULIntervalDay = 7;
	BOOL bCacheInvalid = FALSE;
	BOOL bAddonAllowed = FALSE;

	wchar_t plainwbuf[MAX_PATH] = _T("C:¥¥test¥¥plain.txt");
	wchar_t encryptwbuf[MAX_PATH] = _T("C:¥¥test¥¥encrypt.txt");
	wchar_t decryptwbuf[MAX_PATH] = _T("C:¥¥test¥¥decrypt.txt");
	wchar_t templatewbuf[MAX_PATH] = _T("C:¥¥test¥¥RMSTemplate.xml");

	// Create a rights policy template
	sNameAndDesc.pszName = L"Sample Rights Policy Template";
	sNameAndDesc.pszDesc = L"This is a sample of a rights policy template.";
	sNameAndDesc.lcid = 1033; // LCID(en-us), If you want to set Japanese, you need to set 1041.
	sRightsPolicy.pNameAndDesc = &sNameAndDesc;
	sRightsPolicy.dwNameAndDescNum = 1;

	sUserAndRight.pszUserName = L"user@sample.com";
	sUserAndRight.pszUserId = NULL;
	sUserAndRight.pszUserIdType = ICF_RPT_USER_ID_TYPE_UNSPECIFIED;
	sUserAndRight.dwRight = ICF_RPT_RIGHT_FULLCONTROL;
	sRightsPolicy.pUserAndRight = &sUserAndRight;
	sRightsPolicy.dwUserAndRightNum = 1;

	sRightsPolicy.pExpDate = NULL;
	sRightsPolicy.pbOwnerFullControl = &bOwnerFullControl;
	sRightsPolicy.puiEULIntervalDay = &uiEULIntervalDay;
	sRightsPolicy.pbCacheInvalid = &bCacheInvalid;
	sRightsPolicy.pbAddonAllowed = &bAddonAllowed;
	sRightsPolicy.pAppData = NULL;
	sRightsPolicy.dwAppDataNum = 0;
	sRightsPolicy.pszRefferalURL = NULL;
	sRightsPolicy.pRevocation = NULL;
```

```
	hr = IcfcCreateRightsPolicyFile((ICF_RIGHTS_POLICY*)&sRightsPolicy, (LPCWSTR)templatewbuf);
	if (S_OK == hr)
		printf("IcfcCreateRightsPolicyFile OK!!!¥n");
	else
		printf("IcfcCreateRightsPolicyFile NG(%08x)!!!¥n", hr);

	// File Protection
	hr = IcfcProtect((LPCWSTR)plainwbuf, (LPCWSTR)encryptwbuf, (LPCWSTR)templatewbuf);
	if (S_OK == hr)
		printf("IcfcProtect OK!!!¥n");
	else
		printf("IcfcProtect NG(%08x)!!!¥n", hr);

	//Confirm whether protected file or not
	hr = IcfcIsProtected((LPCWSTR)encryptwbuf, &bRet);
	if (S_OK == hr) {
		printf("IcfcIsProtected OK!!!¥n");

		if (TRUE == bRet) {
			printf("This file has been already protected.¥n");

			hr = IcfcGetContentId((LPCWSTR)encryptwbuf, &guid);
			if (S_OK == hr) {
				printf("IcfcGetContentId OK!!!	ContentID :"
					" {%04x-%02x-%02x-%01x%01x-%01x%01x%01x%01x%01x%01x}¥n",
					guid.Data1,guid.Data2,guid.Data3,
					guid.Data4[0],guid.Data4[1],guid.Data4[2],guid.Data4[3],
					guid.Data4[4],guid.Data4[5],guid.Data4[6],guid.Data4[7]);
			}
			else
				printf("IcfcGetContentId NG(%08x)!!!¥n", hr);

			hr = IcfcGetRights((LPCWSTR)encryptwbuf, &dwRights);
			if(S_OK == hr)
				printf("IcfcGetRights OK!!! Rights:(%08x)¥n", dwRights);
			else
				printf("IcfcGetRights NG(%08x)!!!¥n", hr);

			hr = IcfcUnprotect((LPCWSTR)encryptwbuf, (LPCWSTR)decryptwbuf);
			if (S_OK == hr)
				printf("IcfcUnprotect OK!!!¥n");
			else
				printf("IcfcUnprotect NG(%08x)!!!¥n", hr);
		}
		else
			printf("This file is non-protected. Ready to protect.¥n");
	}
	else
		printf("IcfcIsProtected NG(%08x)!!!¥n", hr);

	return 0;
}
```

・サンプル 2

以下のサンプルでは、マルチデバイス形式による ファイルの保護(IcfcProtectMultiDeviceFormat)、保護形式確認(IcfcIsMultiDeviceFormatFile)、保護解除(IcfcUnprotectMultiDeviceFormat)、使用したメモリの解放(IcfcFreeMemory)という一連の処理をおこなっています。

```
#include "pch.h"
#include <Windows.h>
#include <tchar.h>
#include <iostream>
#include "icfcprotector.h"
#include "IcfRptMaker.h"

#pragma comment(lib, "icfcprotector.lib")

int main(int argc, _TCHAR* argv[])
{
	BOOL bRet = FALSE;
	HRESULT hr;
	DWORD dwFormat = 0;

	wchar_t plainwbuf[MAX_PATH] = L"C:¥¥test¥¥plain.txt";
	wchar_t multioutputwbuf[MAX_PATH] = L"C:¥¥test";
	wchar_t templatewbuf[MAX_PATH] = L"C:¥¥test¥¥RMSTemplate.xml";

	wchar_t policynamebuf[MAX_PATH] = L"Sample Rights Policy Template";
	wchar_t policydesceiptionbuf[MAX_PATH] = L"This is a sample of a rights policy template.";
	wchar_t usernamebuf[MAX_PATH] = L"user@sample.com";

	wchar_t* pOutputFilePath = NULL;
	wchar_t* pDecryptFilePath = NULL;

	// File Protection (MultiDevice)
	hr = IcfcProtectMultiDeviceFormat((LPCWSTR)plainwbuf, (LPCWSTR)multioutputwbuf,
(LPCWSTR*)&pOutputFilePath, (LPCWSTR)templatewbuf);
	if (S_OK == hr) {
		printf("IcfcProtectMultiDeviceFormat OK!!!¥n");
		printf("IcfcProtectMultiDeviceFormat output file to %ls!!!¥n", pOutputFilePath);

		//Confirm whether protected file or not
		hr = IcfcIsMultiDeviceFormatFile((LPCWSTR)pOutputFilePath, &bRet);
		if (S_OK == hr) {
			printf("IcfcIsMultiDeviceFormatFile OK!!!¥n");

			if (TRUE == bRet) {
				printf("This file has been already protected.¥n");

				hr = IcfcUnprotectMultiDeviceFormat((LPCWSTR)pOutputFilePath,
(LPCWSTR)multioutputwbuf, (LPCWSTR*)&pDecryptFilePath);
				if (S_OK == hr) {
					printf("IcfcUnprotectMultiDeviceFormat OK!!!¥n");
					printf("IcfcUnprotectMultiDeviceFormat output file to %ls!!!¥n",
pDecryptFilePath);

					// Memory release
					IcfcFreeMemory((LPCWSTR)pDecryptFilePath);
				}
				else
					printf("IcfcUnprotectMultiDeviceFormat NG(%08x)!!!¥n", hr);

			}
```

```
                        else
                                printf("This file is non-protected. Ready to protect.¥n");
                        // Memory release
                        IcfcFreeMemory((LPCWSTR)pOutputFilePath);
                }
        }
        else
                printf("IcfcProtectMultiDeviceFormat NG(%08x)!!!¥n", hr);

        return 0;
}
```

・ **サンプル 3**

以下のサンプルでは、ファイルフォーマット取得（IcfcGetFileFormat）をおこなっています

```
#include "pch.h"
#include <Windows.h>
#include <iostream>
#include "icfcprotector.h"

#pragma comment(lib, "icfcprotector.lib")

int main(int argc, char* argv[])
{

        if (argc != 2) {
                printf("Illegal argument error");
                return 1;
        }

        HRESULT hr;
        DWORD dwFormat = 0;
        char* pCheckedFile = argv[1];

        // Format check
        hr = IcfcGetFileFormat((LPCWSTR)pCheckedFile, (DWORD*)&dwFormat);
        if (S_OK == hr) {
                printf("IcfcGetFileFormat OK!!!¥n");

                if (dwFormat == 1)
                        printf("This file is IRM Office/FileShell type");
                else if (dwFormat == 2)
                        printf("This file is MultiDevice type");
                else if (dwFormat == 4)
                        printf("This file is NFP type");
                else if (dwFormat == 16)
                        printf("This file is Microsoft compatible/FileShell type¥n");
                else if (dwFormat == 32)
                        printf("This file is Microsoft compatible type¥n");
                else if (dwFormat == 0)
                        printf("This file is non-protected");
        }
        else {
                printf("IcfcGetFileFormat NG(%08x)!!!¥n", hr);
        }

        return 0;
}
```

第4章 FileShell クライアント API の展開

4.1 FileShell クライアント API の展開

FileShell クライアント API は、以下の手順で展開してください。

Operation

FileShell クライアント API のモジュールを、利用するクライアントの任意のフォルダーにコピーしてください。本書では、「C:¥icfadmin¥ClientAPI」にコピーしたと仮定します。

■ フォルダー構成

フォルダー名	説明
¥Tools	
└Develop	
└ClientAPI	
├Include	C/C++開発用ヘッダファイル格納フォルダー
├Lib	C/C++開発用インポートライブラリ格納フォルダー
│ ├x64	64bit 用インポートライブラリ
│ └x86	32bit 用インポートライブラリ
└Module	FileShell クライアント API モジュール格納フォルダー
├x64	64bit 用モジュール
└x86	32bit 用モジュール

■ 配置イメージ

［C:¥icfadmin¥ClientAPI］フォルダー配下に、対応する FileShell クライアント API モジュールを配置します。

```
C:¥icfadmin¥ClientAPI
                └¥IcfcProtector.dll
```

以上で、モジュールの展開は、終了です。

Notice

- 64bitOS 上で 32bit アプリケーションを動作させる場合は、32bit アプリケーション用のモジュールをご利用いただく必要があります。

4.2 権利ポリシーテンプレートの配置と保護時のパス指定

FileShell クライアントAPIを使って開発したアプリケーションが利用する権利ポリシーテンプレートの配置と保護時のパス指定について説明します。

4.2.1 Office IRM/FileShell 形式またはマルチデバイス形式で保護をおこなう場合

Office IRM/FileShell 形式またはマルチデバイス形式の保護をおこなう場合、権限情報が記述された権利ポリシーテンプレートファイルの配置場所は任意です。配置場所を API で引数として指定する際は、ファイル名およびドライブ文字を含む絶対パスを指定します。

＊ RMS サーバーで発行された権利ポリシーテンプレートを使用する場合は、%localappdata%¥Microsoft¥MSIPC¥UnmanagedTemplates にも同じファイルを配置しておく必要があります。ローカル権利ポリシーテンプレートを使用する場合は、上記フォルダー(UnmanagedTemplates フォルダー)には配置しないでください。

＊ FileShell クライアントAPI で%localappdata%¥Microsoft¥MSIPC¥UnmanagedTemplates に権利ポリシーテンプレートを配置し利用する場合、FileShell ポリシーで「クライアントで RMS サーバーから権利ポリシーを取得できるようにする」設定は有効にできません。(有効にした場合、FileShell が正常に動作しない場合があります。)

参照 権利ポリシーテンプレートの作成方法は『FileShell SDK 利用ガイド』の「権利ポリシーテンプレートの準備」を参照してください。

4.2.2 NFP 形式で保護をおこなう場合

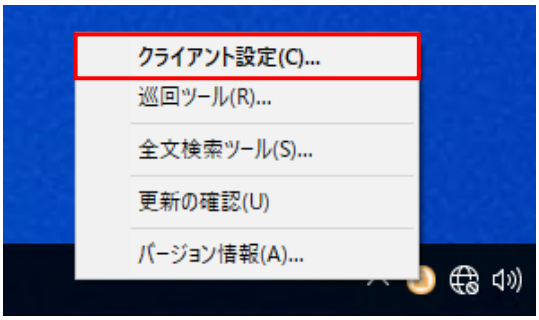
NFP形式の保護をおこなう場合、権限情報が記述された権利ポリシーテンプレートファイルは 以下のいずれかに配置されます。

NFP 権利ポリシーのインポート・適用方式	配置されるフォルダー
FileShell サーバーから受信	%AppData%¥NEC¥InfoCageFileShell¥Policy
クライアント設定ツールでユーザーが作成もしくはインポート	%AppData%¥NEC¥InfoCageFileShell¥EncPolicy
FileShell クライアントのインストーラーに添付され、インストール または IcfPolicyUtil.exe からインポート	%AllUsersProfile%¥NEC¥InfoCageFileShell¥Policy

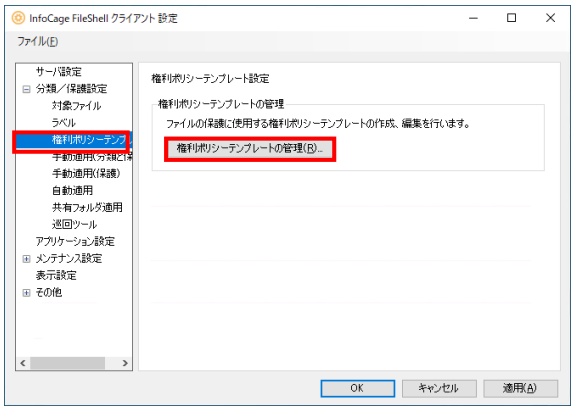
配置場所を API で引数として指定する際は、以下の手順で使用する NFP 権利ポリシーテンプレートを特定し、そのファイル名およびドライブ文字を含む絶対パスを指定します。

Operation

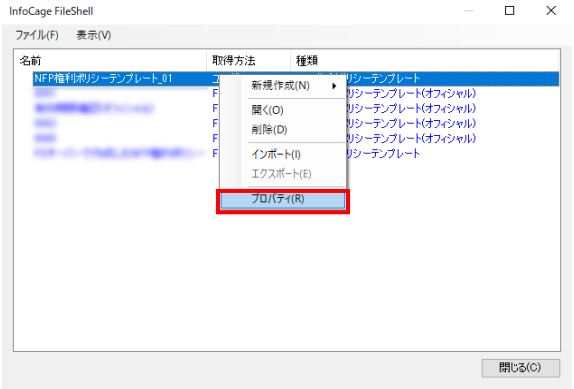
1. 保護に使用する NFP 権利ポリシーテンプレートがインポートされた環境で、FileShell クライアントのクライアント設定を起動します。



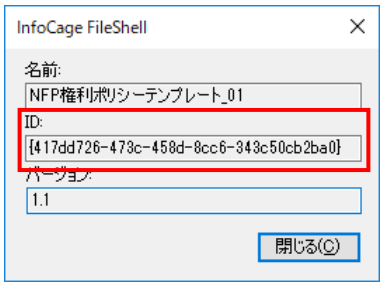
2. 左部メニューより、「分類/保護設定」-「権利ポリシーテンプレート」を選択し、「権利ポリシーテンプレートの管理」ボタンをクリックします。



3. 権利ポリシーテンプレート管理画面が表示されるので、保護に使用する権利ポリシーテンプレートを選択し、右クリックメニューから［プロパティ］を選択します。



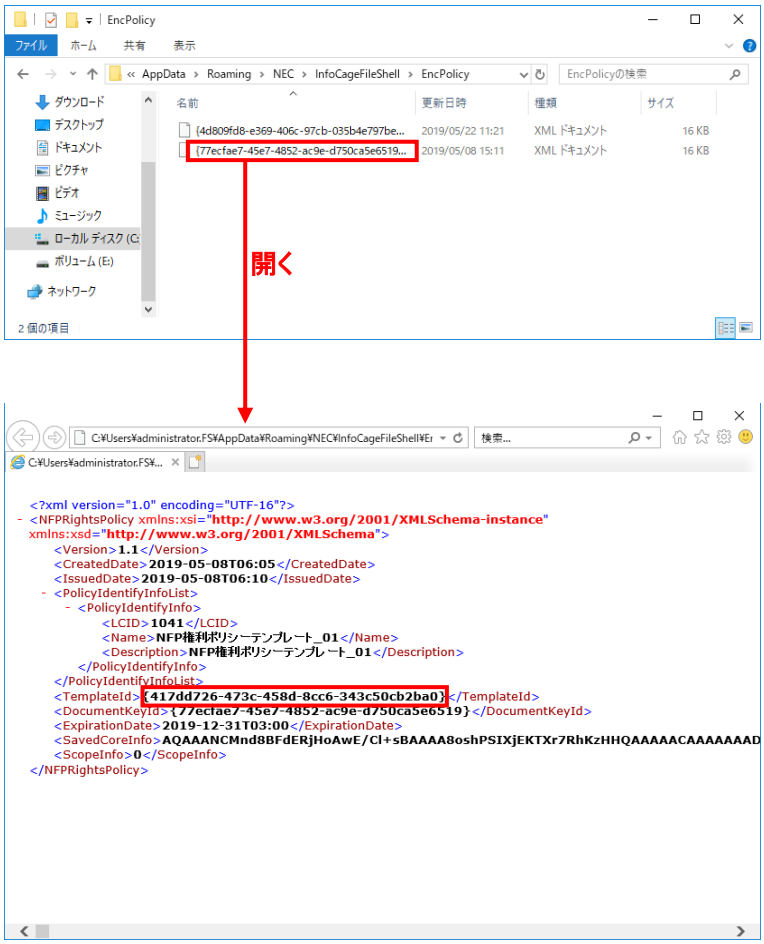
4. プロパティに表示される ID を確認します。



5. エクスプローラーで NFP 権利ポリシーが配置されるフォルダー(※4.2.2 冒頭の表を参照)に格納されている権利ポリシーテンプレート(XML ファイル)を開き、手順 **4.**で確認した ID が<TemplateId>タグに設定されているファイルを探します。手順 **4.**で確認した ID と <TemplateId>タグの値が一致する xml ファイルが、保護に使用する権利ポリシーテンプレートのファイルです。

＊ 格納されている xml ファイルのファイル名は、手順 **4.**で確認した ID ではありません。xml ファイルを開いて<TemplateId>タグの値を確認してください。

＊ 複数のユーザーで共通の NFP 権利ポリシーテンプレートを使用する場合、NFP 権利ポリシーテンプレートが格納されるフォルダーは各ユーザーで異なりますが、ファイル名は同一のものになります。


EncPolicy
ファイル
ホーム
共有
表示
« AppData › Roaming › NEC › InfoCageFileShell › EncPolicy
EncPolicyの検索
ダウンロード
デスクトップ
ドキュメント
ピクチャ
ビデオ
ミュージック
ローカル ディスク (C:
ボリューム (E:)
ネットワーク
名前
更新日時
種類
サイズ
{4d809fd8-e369-406c-97cb-035b4e797be...
2019/05/22 11:21
XML ドキュメント
16 KB
{77ecfae7-45e7-4852-ac9e-d750ca5e6519...
2019/05/08 15:11
XML ドキュメント
16 KB
2 個の項目
開く
C:¥Users¥administrator.FS¥AppData¥Roaming¥NEC¥InfoCageFileShell¥Er
検索...
C:¥Users¥administrator.FS¥...
<?xml version="1.0" encoding="UTF-16"?>
- <NFPRightsPolicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<Version>1.1</Version>
<CreatedDate>2019-05-08T06:05</CreatedDate>
<IssuedDate>2019-05-08T06:10</IssuedDate>
- <PolicyIdentifyInfoList>
- <PolicyIdentifyInfo>
<LCID>1041</LCID>
<Name>NFP権利ポリシーテンプレート_01</Name>
<Description>NFP権利ポリシーテンプレート_01</Description>
</PolicyIdentifyInfo>
</PolicyIdentifyInfoList>
<TemplateId>{417dd726-473c-458d-8cc6-343c50cb2ba0}</TemplateId>
<DocumentKeyId>{77ecfae7-45e7-4852-ac9e-d750ca5e6519}</DocumentKeyId>
<ExpirationDate>2019-12-31T03:00</ExpirationDate>
<SavedCoreInfo>AQAAANCMnd8BFdERjHoAwE/Cl+sBAAAA8oshPSIXjEKTXr7RhKzHHQAAAAACAAAAAAAD
<ScopeInfo>0</ScopeInfo>
</NFPRightsPolicy>

InfoCage FileShell Ver 6.3
クライアント API 利用ガイド

NEC ソリューションイノベータ株式会社

東京都江東区新木場一丁目 18 番 7 号
TEL(03)5534-2222 (代)

Copyright© NEC Solution Innovators, Ltd. 2021-2023.

NEC ソリューションイノベータ株式会社の許可なく複製・改変等を行うことはできません。