File Vault

Securely store documents encrypted on disk. This solution provides two plug-ins that are used together:

- File Vault Connected System Plug-in
- File Vault Component Plug-in

File Vault Connected System

Overview

Configure this Connected System to securely store documents encrypted at the file-level on disk. This may be used in conjunction with the component plug-in to allow users to secure store and download documents.

The Tink Cryptographic Library is used to handle the encryption and decryption of document. Tink works with keysets that are generated with the `tinkey` command. A key type can be specified when generating a key for your specific encryption requirements.

Properties

Field	Description
Tink JSON Proto Keyset	A Tink Cryptographic Library keyset in plaintext JSON format.
Folder UUID	Uploaded files are stored in this folder.
Username	All files uploaded will be shown as created by this user.

Tink JSON Keyset Generation

To generate a keyset for use with the plug-in, you may use the following example:

1. Install tinkey or download prebuilt binary (easiest).

2. Generate your keyset `tinkey create-keyset --key-template AES128_GCM --out-format json --out aead_keyset.json`
    i. You may change the key template that meets your security requirements.
    ii. Additional key templates may be viewed with `tinkey list-key-templates`. Refer to Tink documentation for further details on keyset generation.
3. Copy the contents of `aead_keyset.json` into the `Tink JSON Proto Keyset` field of the Connected System.

It is critical that `aead_keyset.json` be stored securely - do not email it or leave it on disk unencrypted. You may wish to store this file securely for use later (i.e. for key rotation or recovery purposes).

File Vault Upload Component

Function

`fn!fileVaultUploadField( connectedSystem, onUpload )`
Allows the user to upload one file at a time to the encrypted file vault.

Parameters

Field	Description
connectedSystem	The File Vault Connected System to upload documents to.
onUpload	Returns an array of uploaded documents (currently always a single document, to return multiple in future)

Usage considerations

- onUpload returns a dictionary array where each dictionary contains the following keys:
    - `{name: "my-dummy-image", extension: "jpg", size: 152, id: 1234, uuid: "_a-0000ebbe-5e9c-8000-550d-3aef903aef90_4433"}`
- File upload limit starts at 1MiB and cannot be larger than 10MiB.
- You must store the document `uuid`, `name` and `extension` for use later with the File Vault Viewer Component.

- It is the designers responsiblity to clean up unused documents. The recommendedation are:
    - Treat the connected system's folder as a temporary folder and move documents to another folder after uploading.
    - Delete documents more than 24-hours old that still exist in the original temporary folder.
- Anti-virus scanning is performed by the originating clients device.

File Vault Link Component

Function

`fn!fileVaultLinkField( connectedSystem, document, color )`
Displays a link to a single encrypted documents and allows the user to securely download it.

Parameters

Field	Description
connectedSystem	The File Vault Connected System to download documents from.
document	The document to be downloaded. A dictionary with keys for `uuid`, `name` and `extension`

File Vault Viewer Component

Function

`fn!fileVaultViewerField( connectedSystem, documents )`
Displays a list of encrypted documents in a grid and allows the user to securely download them.

Parameters

Field	Description
connectedSystem	The File Vault Connected System to download documents from.
documents	The documents display. An array of dictionary where each dictionary has keys for `uuid`, `name` and `extension`