Client encryption starter kit

Start with a harmless text file. Choose who may see the original before choosing a tool.

The studio made a file for me

We agree on a protected way to send it and verify who will receive it. The studio can still see the original file it created.

Only I may see the real data

We build with a made-up sample. You run the finished tool locally, using passwords you create and keep. The studio receives neither your real records nor their keys.

Before you lock real files

There’s no ‘Forgot my password’ button for these encrypted files. If you lose the only working password or key, Oathkeeper can’t unlock them for you.

Keep a separate, safe recovery copy of what unlocks each file. A backup protected by the same forgotten password won’t help. Reopen a harmless file **from its backup** before deleting any original.

Find the instructions you need

Recipient verification
Check whose key will open the delivery.

7-Zip
Pack and open an encrypted delivery.

KeePass 2
Keep unique passwords in a local vault.

VeraCrypt
Work inside a locked file container.

Aegis Authenticator
Use on Android. On iPhone, choose Apple Passwords or 2FAS Auth.

Proton VPN Free
Protect a connection on shared Wi-Fi.

Proton VPN Free for shared Wi-Fi

If your workplace, network and platform allow it, choose the free plan and the official app for your device. Open it, select **Connect**, and wait for the app to show a connected server. The free plan has unlimited data and permits one active device connection at a time.

The VPN encrypts traffic between your device and its server. Keep using HTTPS and file encryption. A site you sign into can still recognize you.

Send a file to the right person

Agree on the recipient and a contact route you already trust. An email address or a padlock in your browser alone cannot establish who owns an encryption key.

How the proposed local app would work

1. One app on your device would create a pair of keys. You share the **public key** with the studio. The **private key** stays with you and opens the encrypted file. The studio already knows the file it created.
2. Before sending a file, compare the public key's full identifying code, called its **fingerprint**, with the studio in person or over a separate, previously trusted contact route. If it differs or the key changes, stop and verify again.
3. Use the same app to back up your private key in protected form. Keep its unlock passphrase separately. Test restoring the key and opening a harmless encrypted file before relying on it. If the only working copy is lost, we cannot unlock that delivery for you.

The studio is testing this design. A proposed optional authenticator check would control who can download a file. Your private key would still be needed to open it.

Pack the file in 7-Zip for Windows

1. Use Notepad to save a harmless sentence as `practice.txt`. In **7-Zip File Manager**, select the file and click **Add**. Give the archive a neutral name such as `delivery.7z`. The outer name remains visible.
2. Set **Archive format → 7z**. In **Encryption**, enter and confirm a fresh random password, such as a 24-character password generated in KeePass. Select **AES-256** and **Encrypt file names**. Click **OK**. Save the password in your vault.
3. Close and reopen the archive in 7-Zip. You should be asked for the password before seeing filenames. Click **Extract**, choose a new private folder, then open the recovered file and read the practice sentence.

If you agree to a password-protected archive

Send the tested archive through the agreed file channel. Give its unique password to the named recipient through a separate, independently verified private channel that the project permits. If none is available, pause. Ask the recipient to open and check the file. Extracted copies are readable on their device.

You didn't register that device? Contact the studio through an established route and suspend downloads. Verify your identity before enrolling again. Check mailbox sessions, devices and forwarding rules. Change the password if compromise is possible.

7-Zip download · 7-zip.org 7z encryption

Keep the real data with you

Use a trusted computer you control. Set up your passwords privately with screen sharing, recording and remote support turned off.

Let us build from a sample

1. Make a fresh file with the same column names and invented rows. For example, a fake contact can have a fictional account number and an email ending in `@example.com`. Review the whole sample, including hidden sheets and notes.
2. Send only that sample and the result you want. Ask for a local tool that works offline without uploading inputs. Try it first with the sample and the network disconnected.
3. Run the agreed tool yourself on the real records in your private workspace. Save the output there. Share a made-up result or a description if you want our help.

> Keep real files, passwords, recovery keys and screenshots of them out of our chats, screen shares and AI tools. Anything you send to the studio in a private message can still be read by its recipient.

Set up your KeePass 2 password vault

1. On your agreed Windows device, open **KeePass 2.x → File → New**. Create `Private.kdbx` in a private folder. For this setup, choose **Master password** alone. Use a unique, long passphrase, such as six independently random words. Keep its recovery copy in a secure place you control, separate from the vault file.
2. Accept the database settings. Right-click the entry list, choose **Add Entry**, and name it **Practice**. Keep KeePass's generated password and put a harmless reminder in **Notes**. Click **OK → Save**.
3. Close KeePass. Copy the saved `.kdbx` to a separate backup device you control. In KeePass choose **File → Open**, select that backup, type your passphrase and read the practice note. Refresh the backup after changes.

KeePass 2 download · keepass.info First steps Master key and recovery

Make a private workspace

A VeraCrypt container is one locked file. When you open it, Windows gives it a drive letter. You can work in that drive until you close it.

> **Start with a new, empty file container**
> Create a name that does not exist. Selecting an existing file here can overwrite it. If a wizard screen names a disk, partition or system drive as the target, cancel and start again.

Create and open it

1. Open **VeraCrypt → Create Volume → Create an encrypted file container → Standard VeraCrypt volume**. Click **Select File**. In your private folder, type a new filename such as `Practice.hc`.
2. Keep the default encryption settings. For the practice file choose **64 MB**. Create a fresh, long passphrase and save it privately. Keep the ordinary filesystem setting. On the format screen, move the mouse as prompted. Confirm that the target is your *new file*, then click **Format → Exit**.
3. In VeraCrypt choose an unused drive letter, **Select File → Practice.hc → Mount**, then enter your passphrase. In File Explorer open that letter. Copy a harmless text file into it and read it.
4. Close the file and its windows. Select the volume in VeraCrypt and click **Dismount**. Mount it again and read the file. While mounted, people or programs using your unlocked computer can read its contents.

Prove you have a working backup

Create a second new container on a separate trusted drive, at least as large as the first. Leave **Quick Format** and **Dynamic** off. Mount both at different letters, then copy the practice file directly between them. Dismount both. Mount just the backup, open the practice file, then dismount it. Keep the backup’s unlock material separately.

Creating a container leaves existing originals and any cloud copies readable. Use the private drive for new sensitive files. Whole-drive encryption is a separate, optional setup tailored to your device.

VeraCrypt download · veracrypt.io Container walkthrough Backup method

Protect your sign-ins

Choose a protected authenticator

Android: we recommend Aegis. **iPhone:** use Apple Passwords if you already keep logins there (iOS 18 or later). Choose 2FAS Auth if you prefer a separate app. Use a strong device lock and the app's own lock where available. Cloud encryption alone tells you little about how an app protects codes on an unlocked phone.

Practice in Aegis on Android

Aegis creates changing sign-in codes for a service that supports an authenticator app. Set up its recovery before relying on it for a real account.

1. Install **Aegis Authenticator** from its official site. During setup, choose a **password-encrypted vault**. Use a unique passphrase and keep a separate safe recovery copy. Fingerprint unlock can be added for convenience.
2. First add a dummy token. Tap **+** → **Enter manually**. Give it the name **Practice**, leave the time-based defaults, and use the public practice key `PRACTICEPRACTICE`. This sample belongs to no account.
3. In **Settings** → **Import & Export** → **Export**, choose **Aegis (.JSON)** with **Encrypt the vault** enabled. Save a copy on a private device or drive away from the phone. Keep its unlock password separately, including a separate export password if you set one.
4. Using an empty Aegis vault on a second trusted Android device or Android user profile, choose **Settings** → **Import & Export** → **Import from file** → **Aegis**. Open the backup with its password. With both clocks on automatic time, check that the two **Practice** codes match in the same time window.
5. For a real account, open the service's own **Security** → **Two-factor authentication** → **Authenticator app**. On a private screen, display its setup QR. In Aegis tap **+** → **Scan QR code**. Enter the current code *on that service* and wait for confirmation. Keep the service's recovery codes somewhere safe outside Aegis. Refresh your encrypted backup after adding accounts.

Keep the setup secret private. Avoid unencrypted exports and screenshots or photos of a setup QR or key. Someone holding a copy can generate future codes until you revoke or replace that enrollment at the service.

> A service's recovery codes can restore access to that service. They cannot open your Aegis, KeePass or VeraCrypt vaults or a 7-Zip archive.