

Encryption

Keep it secret, keep it safe.

--Gandalf

- [Protected Messages](#)

Protected Messages

Sometimes you want to transmit a message (text, a file, whatever) in such a way that only *you* and *the recipient* know what's in the message and can access its contents. Use cases may include:

- Sharing a password with someone.
- Sending a love letter you want to be private.
- Health care providers exchanging private documents (with each other or with patients).
- Contractors exchanging proprietary knowledge with their business partners.
- Telling someone the secret location of your buried treasure.
- etc.

The thing is, you don't always have a secure channel to transmit this information. So this guide is meant to give you some options.

What we're talking about is *Encryption*: obscuring digital information so that it can only be read in its true form once it is "decoded" by some method, like a password, key, etc.

Secure Options

So what are some options for sending a secured message? Let's look at a few useful tools...

Email: PGP

[What is PGP Encryption?](#)

This section is still being written.

Private Links

There are some tools out there to let you deliver some text or a file attachment in the form of a simple URL you send to someone else.

PrivateBin

[PrivateBin](#) let's you enter any normal text, format it as plain, source code, or markdown, and generate it as a URL to send to anyone. You can set it to expire after a certain period (so it totally vanishes from history), select it to destroy itself after the first time it's read, and password protect it. You can also attach files and include the option to make an open (anonymous) discussion of the content. You can email the link from within the tool, or generate it as a QR code.

Scrt.Link

[Scrt.Link](#) is very similar; it lets you enter text or upload a file with a message, either of which can be password protected and then generated as a URL. It also offers a "Neogram" which is its version of a self-destructing link. While PrivateBin is entirely anonymous, you can opt for a free account with scrt.link to enable features like email or [ntfy](#) read receipts, larger content restrictions, a Slack app and browser integrations, and emoji links.

Send

[Send](#) is a simple tool to upload a file (or several) and generate a link for someone to download it later. You can set an expiration by time or number of downloads allowed, and set a password. It's simple and quick to use.

Encrypt Files with Hat.sh

[Hat.sh](#) is a nifty tool to encrypt any kind of file. Drag-and-drop, add a password or public key, and download the encrypted file (it'll have the same file name with `.enc` added to the extension). The best part is; this whole process happens entirely *in-browser*, meaning the file never leaves your device and goes to a server. Whoever you give the file to simply goes to the same site to decrypt it again, *also* entirely in-browser, and you're done!

Encrypt Content with PrivacyProtect.dev

[PrivacyProtect.dev](#) will let you enter a message **or** a file (not both) that is password protected and include a little password hint. Then it'll encrypt it for you and give you a `.html` file download. Send this to anyone, and when they open it up they'll see the hint, enter the password, and have the content! Once again, the data you're encrypting *never leaves your device*.

What's great about this is that you can send it through any medium you like, you don't have to agree on a password before hand (or even have planned who the receiver will be), and, since it's stored as an `HTML` file, it can be read on *any* device, computer, phone, tablet, etc. Oh, and did I mention you can decrypt it entirely offline? No need for a connection!

On Passwords

Most of the options you'll find employ the use of a password that only you and your recipient know to protect the information. From a security standpoint, it is advised that you **exchange this password via a different channel than the one you are transmitting the message** or else **agree on a password beforehand another way**. This forces a potential leak to be in two places in order to break your system, which is much harder to do.

Good example: You are working with a client's sensitive data and are about to send a confidential report via email. You encrypt the file with a password you agreed to over a private messaging service to avoid including the password along with the file in the same email.

Bad example: You SMS someone a link containing a protected document and then also text them the password to open it. Now anyone who gains access to the text messages (or even just looks over the recipient's shoulder) can *also* access that file.

Think of it this way: If you attached your address to your key ring and then somebody got a hold of your keys, it'd be a lot easier to return the keys to you, but it'd *also* be a lot easier to use them against you.

On Encryption

There are many different kinds of encryption, but for the sake of brevity I only want to point out E2EE, as it's used in most of the tools I recommend. E2EE, or [End-to-End Encryption](#), means that something is encrypted from sender to receiver and at every step in between so that nobody but the sender and the receiver can decode it.

When you use email, text/SMS, or one of the tools I'm about to detail, the service provider handles the data you're transmitting. E2EE means that your message is encrypted *before* that provider gets it and isn't decrypted until *after* they've handed it off to the recipient.

End to End Encryption

The diagram illustrates the flow of data in an end-to-end encrypted system. It features a central blue server icon with a lock, representing the data storage or processing hub. On the left, a green user icon sends data (represented by a green speech bubble with a lock) to the server. On the right, an orange user icon receives data (represented by an orange speech bubble with a lock) from the server. The server also stores data (represented by a blue server icon with a lock). The flow is indicated by arrows: a green arrow from the green user to the server, an orange arrow from the server to the orange user, and a blue arrow from the server to the blue server icon. The server icon has a lock, indicating that the data is encrypted at rest.

Think of it like sending a letter in the mail. If that letter gets intercepted at any point (or opened by the postal worker!), someone else gets to read it. However, if you wrote the letter in a language that only you and the recipient speak/read, then it's a lot safer!