secret in

A team-oriented open source password manager with a focus on transparency, usability and security.
Who am I ?

Florian Gaultier

Security engineer in charge of SCRT France

I break things for a living, find vulnerabilities in software built by others

I’m also a CTF player with 0daysober team
3rd place DEFCON (Las Vegas), 2nd Codegate (Seoul)

I organize Sthack with friends in Bordeaux !
Why this name?

I won a *.me domain name thanks to 15th Gandi’s anniversary

Thought about the joke secret-in.me

Needed something to host on it

Decided to develop a password manager

(I also own https://so.much.beer, you’re welcome)
What’s a password manager?
Why do we need a password manager?

A password is the lock on your door
Keep your private data... private

Must have one password by field to prevent one compromised website to give your only password to the world

Besides, the more a password travels, the more you should change it (increased compromise probability)

Impossible to remember hundreds of changing passwords
Databaseless password manager

Use a pure algorithm which basically computes a password from ("website"|"secret").

**PROS**
You don’t need any storage nor synchronisation process

**CONS**
You can’t easily change a password
You can’t comply with weird password policies

WHAT'S A PASSWORD MANAGER?
Password manager with database

Generate one password by field and store it physically in a notebook

**PROS**

- Need physical access to steal passwords
- Easy to use
- Nothing to memorize

**CONS**

- You need to keep it with you
- Hard to backup, hard to update
- Copy/paste doesn't exist in physical world
WHAT'S A PASSWORD MANAGER?

Password manager with digital database

**PROS**
Use algorithms and computers to store your passwords
One secret **master password** to lock all the others in a safe encrypted database

**CONS**
Trust in software and technologies
Your master password is the **SPOF** (Single Point Of Failure)
Password managers: State of the art
Proprietary software

Example: Lastpass, 1Password, Dashlane...

**PROS**
- A lot of features
- Multiple devices support
- Enterprise support

**CONS**
- Vendor-lockin
- Blindly trust the vendor
- Costs money (what happens if they raise their prices)
WHY ANOTHER PASSWORD MANAGER?

Open source software

Example: Keepass, Passbolt...

**PROS**
Auditable by anybody
No vendor-lockin, free like free speech
Self-hosting

**CONS**
No great support
Not so great UX-UI
Life of a pentester

No clear path for companies.

**Keepass** not designed to be shareable is used with weird SMB synchronisation mechanisms

Open source or private weird solution with **LDAP** binding!

LDAP binding (or SSO) is like using the same password everywhere...
Yes but secret-in
Development goals

No heavy software

Upgrade mechanisms, executable to trust…

Never roll your own Crypto

Writing crypto is hard, like really hard!

Built for companies

Open source may scare companies
No heavy software

One thing you have on almost any device: Browser

Secret-in core only uses JavaScript

“Wait, what? You wrote crypto in JavaScript?!”
Never roll your own Crypto

W3C produced WebCryptoAPI spec (out of draft in February 2017)

Contains standard cryptographic algorithms (hash, asymmetric, symmetric)

Built in browser engine so it's not JavaScript

“You trust Google/Apple/Microsoft engineers don’t you?”
Built for companies

Trust and transparency

WebApp code splitted between **simple core lib** and UI-UX wrapping

Core lib contains the logic and can be "easily" audited.

```
secretin-app
```

```
secretin-lib
no dependencies
```

React
Electron
Built for companies

Sharing capabilities
Read, Read/Write, Read/Write/ReShare

Traceability
Who, When, What

Documentation
Self-hosting made easy

Good UI-UX
UX engineers are now part of the project
Offline mode

Everything client side

Malicious access to the database must not compromise anything, not even metadata (not even username)

No logic layer, only use crypto to achieve good confidentiality

RESULT

Classic password manager features plus sharing capability with big blob of JSON to copy paste as your encrypted database.
Synchronisation

Introduce a **server** to store the encrypted database

Server can’t compromise confidentiality, nor can the network

Server can introduce a logic layer:

- Authenticate to give encrypted database to legit users.
- Add **granularity** in sharing process (Read only, read/write, read/write/reShare)

RESULT

Synchronisation with authentication
More protections

Server means anonymous access attempt

Add bruteforce detection (by IP address)

Add 2 Factor Authentication (with Google Authenticator)

RESULT

Encrypted database is well protected
STEP BY STEP

More usability

Type your long master password plus 2FA is annoying

Introduce trustable device feature

Shortpass plus trusted device unlock your key

RESULT

Fast login with good security
STEP BY STEP

The return of the offline mode

A desktop application adds offline synchronisation feature

Based on Electron to wrap secretin-app (reused codebase)

Saves a local database backup to access it offline

RESULT

Cross platform application with offline synchronisation
Password manager: technical-boring stuff
How does it work?

Cryptographic level guarantees confidentiality

- Classic RSA 4096 asymmetric usage to share intermediate key
- Intermediate key encrypts secret with AES-256
- Your private RSA key is encrypted with a derived form of your master password PBKDF2(SHA-256)

Logic level adds more confidentiality and features

- Stateless requests are signed by user private key
- Server verifies the signature then the rights on the claimed secret access (with anti-replay mechanism)
Technologies

Everything is JavaScript

Use simple CouchDB database, easy to replicate and scale

https://secret-in.me static content on GitHub

https://api.secret-in.me hosted on IBM Bluemix

CouchDB on IBM Cloudant
Wrap up
Tradeoffs and limitations

WebCryptoAPI is **young**
Very few compatible browsers (only works on Chrome and Safari on iOS 11)

Crypto **takes time**
Particularly slow on mobile browser (~x5 slower)

No god mode
You only control your own data
Features Summary

Create/Update/Delete a **Secret**

**Share** with permissions (Read, Write, ReShare)

Folders to organize your secrets

2 Factor Authentication (by token or by device with shortpass)

**Offline** Mode (with non-shared secrets editable)

**Export/Import** between secret-in instances

Lib v2 (out last week) adds **nodeJS adapter** (based on node-forge) to be able to build bots
WRAP UP

Coming Next

SOON
Secret history
Trace access
UI/UX improvement
Documentation improvement
Institutional website

NOT SOON
Native mobile application
Browser extension
Import from other password manager (only from KeePass for now)
How to get it?

Test it: [https://secret-in.me](https://secret-in.me)

GitHub: [https://github.com/secretin](https://github.com/secretin)

Self-Host:


[https://github.com/secretin/secretin-app#setup-the-app](https://github.com/secretin/secretin-app#setup-the-app)

Twitter: @agixid, @antoinelyset, @calyhre, @dqms_output, @vcent_ricard

My contact: florian@scrt.fr