Open source password manager for teams
Throwback from the past
Passbolt team at Fosdem in 2017
Who is using a password manager?
Wait another password manager?

or how is Passbolt different from...
Based on **OpenPGP** and **public key cryptography**.

Standing on **OpenPGP.js**, **GopenPGP** & **Gnupg** shoulders

+ No “master key” derived from the user password using PBKDF2 (*warden, Lastpass), or Argon2 (Keepass, Dashlane).

+ Encryption strength is not dependent on user password strength or number of derivation rounds.

+ Interoperability and support for new/future algorithms (Curve25519).

- Harder for the end-user, they must not lose their private key nor their passphrase.

- Similarly, requires complex key transfer protocols (e.g. QR codes).
Signature + challenge authentication

Using gpgauth principles with a twist.

+ Not prone to credential stuffing.
+ Not prone to phishing.

- More complex to implement for developers. Not a standard (yet 😐)
Mandatory browser extension

JavaScript cryptography considered harmful? (the return)

+ If server is compromised an attacker cannot add javascript to the login page to extract the master password (and secret key if any).

+ Using the background page sandbox allows to isolate decrypted private key (reduced XSS impact).

+ Automatic rollout of signed updates for all clients in case of cryptography issues.

+ Extension is tied to run only on a trusted domain. In an iframe.

+ Bonus: quick access / form integrations possible.

- Not useful when attacker has read access on the client OS. (unlike keepass)

- Need to trust firefox, etc. stores. Or build and host your own.

- 3rd party websites can find out extension is installed.
Anti-phishing passphrase protection by default.

“Something” selected by the user only known by the client.

+ Protect every sensitive operations against phishing attacks.

+ Interactive, prevent an attacker to put “one more layer on top”.

- The majority of users interviewed do not understand the concept of phishing in general.

~ More services are warming up to this (banks, competitors, etc.).

An anti-phishing token appears each time an end-user is prompted to enter their master password.

#Friends: Check out mailvelope for similar concepts in the field of email encryption.
Full transparency

Clear about both the strengths & the residual risks.

+ No opt-out telemetry / analytics. No “phoning home”.
+ 100% Open Source. No open core. Yes, even the “pro” offer.
+ Independent 3rd party audits at least once a quarter or for new large feature launches (cure53).
+ Annual SOC2 Type II audited report.
+ Financially stable.

- Server is trusted for public key distribution. No clear key signatures.
- Searchable API = unencrypted metadata.
- No quantum resistant algorithm available yet.
- Export feature is on by default.
- No clear indications when passwords should be rotated.
- No user key rotations
- We’re only humans after all.
How does it look like?
Available on most **browsers and devices**.

Safari support and desktop app coming in 2023
and most terminals.

Via official or community maintained CLIs, libraries, etc. or just Curl and Gpg

✓ Retrieve, store, and share passwords programmatically with the JSON api.
✓ Automate recurring tasks with CLI.
✓ Use with Ansible collection, Gitlab, etc.

```bash
$> export SECRET=`curl --location --request GET '${PASSBOLT_URL}'
--header 'Authorization: ${ACCESS_TOKEN}'
--header 'Content-Type: application/json'
| jq -j '.body.data'
| gpg -q --no-tty
| jq -j '.password`
```

```bash
$> passbolt get 664735b2-4be7-36d9-a9f8-08d42998faf8
-----BEGIN PGP MESSAGE-----
```

```bash
$> passbolt get
$(passbolt find | awk '/server/ { print $NF }')
| gpg -q --no-tty
```
Go ahead share your **secrets**

We won’t tell the others.
Quick access & in form menu

Fast & furious.
Android and iOS

Your credentials on the go.
Got trust issues? Host it yourself!

Works in air-gapped environments or on a Raspberry Pi

- Docker images (rootless / distroless)
- Debian / Ubuntu packages
- RPM packages (RHEL, Centos, etc).
- Helm charts
- AWS AMI, Digital Ocean, etc.
What’s cooking?

Q1/Q2 2023 Community Roadmap

✓ Mobile to mobile key transfer
✓ Enforce MFA policy
✓ User self-registration (allowed domains)
✓ Passkeys (Webauthn) support for 2FA
✓ Grid configuration & improvements
✓ New help site
✓ TOTP on mobile devices
✓ Desktop application

& beyond...

✓ Manifest v3
✓ Passwords expiry
✓ Custom fields / more content types
What is it made of?

(the secret ingredient is love)
Three main application layers.

- **Styleguide, Browser Extension, Web API**

**User Interface**
Interactions, Styling

**Sensitive operations**
Crypto, validation, API calls

**JSON API**
(Some HTML content)

**Baseline services**
LAMP or similar

---

Webextension / Styleguide (React / Less)

Webextension background Page (JS)

Passbolt API (Cakephp / PHP)

OpenPGP.js

GnuPG

Nginx / Apache  
File / Redis / etc.  
Mariadb / Postgresql / Etc.

GNU/Linux
Web application layer

https://help.passbolt.com/api

View one resource

It is possible to get data for a single resource identified by the unique UUID. All you need is to make a GET request to /resources/<resourceId>.json.

GET /resources/<resourceId>.json

Possible responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
</tr>
<tr>
<td></td>
<td>Response includes the resource metadata object.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request</td>
</tr>
<tr>
<td></td>
<td>The resourceID supplied is not a valid UUID</td>
</tr>
<tr>
<td>403</td>
<td>Authentication Failure</td>
</tr>
<tr>
<td></td>
<td>The user making the request is not authenticated</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
</tr>
<tr>
<td></td>
<td>The resource either does not exist or the user does not have access permission.</td>
</tr>
</tbody>
</table>

Examples

Success response

For another perspective on the API you browse the OpenAPI 2.0 specifications using the dedicated API reference site (Swagger UI).

API Reference

You can also find the latest OpenAPI 2.0 specifications directly on the dedicated repository.

OpenAPI Specs repository
Web application layer

Database tables (main ones)
Web application layer

A day in the life of a HTTP request in a typical MVC application
Browser extension

Manifest v2 and beyond...

Background page

- Main
- Pagemod
- worker
- Port

Content Script / Web accessible script (iframe)

- React App
- React Components

DOM

Controllers

Events

Models

Services

Vendors

openpgpjs, etc.

Local Storage

React contexts

ApiServices

Collections

Entities

Vendors

xregexp, react

Json API

create

insert

DOM
Passbolt styleguide (Storybook)

https://passbolt.github.io/passbolt_styleguide
Thank you Fosdem ❤

See you at the bar at 18:00 for some swag & 🍺