Jumeau numérique · Sous-section

Gérer ses credentials proprement

Organiser et sécuriser tes API keys, tokens et mots de passe. Bitwarden, Vaultwarden, macOS Keychain, bonnes pratiques .env.

Source : https://prisme.one/plateforme/ia-juillet-2026/semaine-3-credentials

Objectif

Séparer l'inventaire des comptes (Infrastructure.md) du coffre des secrets (password manager).

Output attendu

Infrastructure.md à jour, password manager choisi, API keys migrées, .env dans .gitignore.

Le problème des secrets (plus petit qu'on ne croit)

Tu accumules des API keys, des tokens, des mots de passe. Anthropic, OpenAI, Telegram, Cloudron, NocoDB, PocketBase... Au début tu les mets dans un `.env`, puis dans une note, puis dans un message Telegram "pour ne pas oublier".

Avant de sortir l'artillerie lourde, dédramatisons. La dernière promo a perdu un temps fou à installer et configurer Bitwarden. Dans la grande majorité des cas, tu n'en as pas besoin.

Le vrai risque, ce n'est pas d'avoir une clé en clair dans un fichier. C'est qu'elle **fuite vers l'extérieur**. Tant que tes secrets restent dans ton coffre chiffré de bout en bout (Obsidian Sync) et sur ton disque, tu es tranquille.

Les patterns de fuite qui comptent vraiment :

1. Un `.env` commité sur un repo git **public**
2. Une clé collée dans un **chat public** (session ChatGPT partagée, Discord, forum)
3. Un secret affiché par l'IA dans une réponse que tu partages ensuite publiquement
4. Un `.env` envoyé en copier-coller par email ou messagerie non chiffrée

Ce qui n'est PAS un drame : une clé en clair dans `.env`, `opencode.json` ou une note, **tant que ton coffre est chiffré E2E via Obsidian Sync**. C'est le cas par défaut si tu as suivi le bootcamp.

La hiérarchie recommandée

La règle fondamentale : **séparer l'inventaire du coffre**. L'inventaire dit *ce qui existe*. Le coffre contient *les valeurs secrètes*.

1
Inventaire des comptes
Quel service, quel email, quel usage
→ Note markdown (`Infrastructure.md`)
2
API keys / tokens / mots de passe (les VALEURS)
Les secrets eux-mêmes
→ En clair dans ton coffre Obsidian (chiffré E2E), ou password manager si tu veux
3
Variables de dev local
Fichiers `.env` avec les valeurs, dans un projet non public
→ En clair si ton coffre est chiffré E2E, ou référence au password manager

En résumé : `Infrastructure.md` dit "j'ai un compte Anthropic à tel email", sans jamais contenir la valeur. La clé API, elle, vit soit en clair dans ton coffre Obsidian chiffré E2E (le plus simple), soit dans un password manager si tu veux un cran de plus. L'important : elle n'est jamais exposée dans un espace public.

Les niveaux de setup

Choisis selon ta situation. Pour la majorité des participants, le **niveau 0 suffit largement**. Les niveaux suivants sont optionnels, pour ceux qui veulent un cran de plus ou qui gèrent des données client très sensibles.

Niveau 0 : Clés en clair dans ton coffre chiffré E2E (recommandé par défaut)

Niveau 0 : Le plus simple, recommandé
Coffre Obsidian Sync (chiffré E2E)

Zéro installation, zéro CLI à brancher
Tes clés en clair dans `.env`, `opencode.json` ou une note
Chiffrement de bout en bout géré par Obsidian Sync
Ton IA accède aux clés directement quand elle en a besoin

C'est le setup que **j'utilise personnellement**. Je n'utilise pas Bitwarden. Mon coffre est synchronisé avec Obsidian Sync, donc chiffré de bout en bout, et mes clés vivent en clair dedans. Rien ne fuite tant que je ne pousse pas ça sur un repo public ou dans un chat public.

Les 2 garde-fous : `.env` dans `.gitignore`, et jamais de clé dans un espace public. C'est tout.

Niveau 1 : Bitwarden (optionnel, pour centraliser)

Niveau 1 : Optionnel, centraliser tes secrets
Bitwarden
Gratuit, cloud, chiffré de bout en bout
Extension navigateur + app mobile
CLI disponible (`bw`) pour les scripts
MCP server officiel disponible
Setup en 10 minutes

Setup :

1. Crée un compte sur bitwarden.com (gratuit)
2. Installe l'extension navigateur
3. Pour chaque API key, crée une entrée dans Bitwarden avec le nom du service et la valeur

Pour les plus techniques — Bitwarden a un CLI (`bw`) et un MCP server officiel qui permet à ton IA de récupérer des secrets de manière sécurisée :

```
# Installer le CLI Bitwarden
npm install -g @bitwarden/cli

# Se connecter
bw login

# Récupérer un secret
bw get item "Anthropic API Key" | jq -r '.login.password'
```

Niveau 2 : macOS Keychain (Mac users)

Niveau 2 : Pour les utilisateurs Mac
macOS Keychain
Déjà installé sur chaque Mac, zéro config
Commande `security` en terminal
Chiffré par le Secure Enclave du Mac
Gratuit, local, pas de cloud

Stocker un secret :

```
# Ajouter une API key au Keychain
security add-generic-password \
  -a "mon-compte" \
  -s "ANTHROPIC_API_KEY" \
  -w "sk-ant-api03-xxxxx" \
  -T ""
```

Récupérer un secret :

```
# Lire la valeur
security find-generic-password \
  -s "ANTHROPIC_API_KEY" \
  -w
```

Utiliser dans un `.env` :

```
# Script qui génère le .env à partir du Keychain
echo "ANTHROPIC_API_KEY=$(security find-generic-password -s
ANTHROPIC_API_KEY -w)" > .env
```

L'avantage : rien ne quitte ta machine. L'inconvénient : c'est local — si tu as un serveur distant, les secrets ne sont pas synchronisés.

Niveau 3 : Vaultwarden (auto-hébergé)

Niveau 3 : Pour les self-hosters
Vaultwarden
Réimplémentation légère de Bitwarden, auto-hébergée

Installable via Cloudron (1 clic)
Tes données restent sur TON serveur
Compatible avec tous les clients Bitwarden
MCP server Bitwarden compatible

Si tu as déjà un serveur Cloudron (semaine 3), Vaultwarden est disponible dans l'App Store Cloudron en un clic. Sinon, voici le docker-compose :

```
services:
  vaultwarden:
    image: vaultwarden/server:latest
    container_name: vaultwarden
    restart: unless-stopped
    ports:
      - "8080:80"
    volumes:
      - vaultwarden-data:/data
    environment:
      DOMAIN: "https://vault.ton-domaine.com"

volumes:
  vaultwarden-data:
```

Une fois lancé, tu accèdes à l'interface web sur `https://vault.ton-domaine.com`, tu crées ton compte, et tu utilises les **mêmes clients Bitwarden** (extension navigateur, app mobile, CLI) — ils sont compatibles.

Comparatif

Bitwarden
Keychain
Vaultwarden
Prix
Gratuit
Gratuit
Gratuit
Hébergement
Cloud Bitwarden
Local (Mac)

Ton serveur
Multi-device
Oui
Mac seulement
Oui
CLI
bw
security
bw
MCP
Officiel
Non
Compatible
Souveraineté
Moyen
Total
Total
Difficulté
Facile
Facile
Moyen

Ma recommandation : reste au niveau 0. Des clés en clair dans un coffre chiffré E2E (Obsidian Sync) couvrent l'immense majorité des besoins, et c'est ce que j'utilise moi-même. Si un jour tu gères des données client très sensibles ou tu veux centraliser proprement, Bitwarden (niveau 1) est le meilleur point de départ, et la migration vers Vaultwarden reste transparente (mêmes clients, mêmes données).

Le fichier Infrastructure.md

C'est ton **registre** — l'inventaire de tout ce qui tourne, sans les valeurs secrètes. Il vit dans ton vault Obsidian (par exemple `4 TOOLS/Infrastructure.md` ou `2 CASQUETTES/[ta casquette tech]/Infrastructure.md`).

Structure recommandée

```
# Infrastructure

## Serveurs
| Service | URL | Usage |
|---------|-----|-------|
| Serveur principal | 65.xxx.xxx.xx | Cloudron, apps |
| Cloudron | https://my.domaine.com | Panel admin |

## APIs & Services
| Service | Email du compte | Usage | Secret dans |
|---------|----------------|-------|-------------|
| Anthropic | mon@email.com | Claude API | Coffre .env (E2E) |
| OpenAI | mon@email.com | Whisper, GPT | Coffre .env (E2E) |
| Telegram Bot | - | Bot IA perso | Bitwarden |

## Bases de données
| Service | URL | Usage |
|---------|-----|-------|
| NocoDB | https://nocodb.domaine.com | Données clients |
| PocketBase | https://backend.domaine.com | Auth plateforme |

## Domaines
| Domaine | Registrar | Usage |
|---------|-----------|-------|
| mon-domaine.com | OVH | Site principal |
```

Ce qui est dans ce fichier : les noms, les URLs, les emails de compte, les usages. L'IA peut lire ce fichier pour savoir quels services existent.

Ce qui n'est PAS dans ce fichier : les API keys, les tokens, les mots de passe. Jamais. La colonne "Secret dans" indique *où* trouver la valeur (ton coffre `.env`, Bitwarden, Keychain...), pas la valeur elle-même.

Bonnes pratiques `.env`

Règle 1 : `.env` dans `.gitignore`

Toujours. Sans exception. Vérifie maintenant :

```
# Vérifier que .env est ignoré
cat .gitignore | grep -i env
```

Si ce n'est pas le cas, ajoute :

```
# Secrets
.env
.env.local
.env.*.local
```

Règle 2 : Créer un `.env.example`

Le `.env.example` contient les **noms** des variables sans les valeurs. Il est commité sur git — c'est la documentation de quelles variables sont nécessaires :

```
# .env.example — les noms sans les valeurs
ANTHROPIC_API_KEY=
OPENAI_API_KEY=
TELEGRAM_BOT_TOKEN=
NOCODB_URL=
NOCODB_TOKEN=
```

Quand tu installes le projet sur une nouvelle machine, tu copies `.env.example` vers `.env` et tu remplis les valeurs.

Règle 3 : Ne jamais commiter un `.env`

Si tu as déjà commité un `.env` par erreur :

```
# Retirer le fichier du tracking git (sans le supprimer)
git rm --cached .env
git commit -m "remove .env from tracking"
```

Attention : si le `.env` a été poussé sur un remote **public** (GitHub, GitLab), les secrets sont compromis. Change toutes les clés API qui étaient dedans. Sur un remote privé ou chiffré le risque est bien plus faible, mais par précaution, change-les quand même.

Pattern avancé (optionnel) : références au password manager

Uniquement si tu utilises un password manager (niveau 1 ou 3). Au lieu de laisser les valeurs en clair dans `.env`, tu peux les injecter depuis le coffre au moment voulu :

```
# Script setup-env.sh — génère le .env depuis Bitwarden
#!/bin/bash
bw unlock --check || bw unlock

echo "ANTHROPIC_API_KEY=$(bw get password 'Anthropic API Key')" >
.env
echo "TELEGRAM_BOT_TOKEN=$(bw get password 'Telegram Bot')" >>
.env
echo "NOCODB_TOKEN=$(bw get password 'NocoDB Token')" >> .env

echo "✓ .env généré depuis Bitwarden"
```

Tu lances `./setup-env.sh` une fois, et ton `.env` est rempli. Le script peut être commité (il ne contient aucun secret), le `.env` généré ne l'est jamais.

Checklist

- [ ] J'ai créé ou mis à jour mon fichier Infrastructure.md (sans valeurs secrètes)
- [ ] J'ai choisi mon niveau de setup (clés en clair dans coffre E2E par défaut, password manager si besoin)
- [ ] Mes clés API sont dans un coffre chiffré E2E (Obsidian Sync) ou un password manager, jamais dans un espace public
- [ ] Mes fichiers .env sont dans .gitignore
- [ ] J'ai un .env.example avec les noms des variables (sans valeurs)
