SSH-SK (Software Key)

Secure SHell – Software Key....2
La fin du fichier "Clé Privée"....2
Le Mécanisme de Dérivation (Non-Résident)....2
Processus d'Authentification....2
Exception : Les Clés Résidentes (Resident Keys)....2
GÉNÉRATION DE LA CLÉ (BUILD)....3
DÉPLOIEMENT & UTILISATION (RUN)....4
Propagation de la Clé Publique....4
Simplification de la Connexion (Best Practice)....4
Le Rituel de Connexion....5

Secure SHell – Software Key

Note Importante (Pré-requis) : Pour que cela fonctionne, il faut que le **Client** ET le **Serveur** aient une version d'OpenSSH **supérieure ou égale à 8.2**.

Si vous administrez de vieux serveurs (Debian 9/10 ou CentOS 7), cela ne marchera pas nativement. Sur Debian 11/12/13, c'est natif.

```
ssh -V
```

```
OpenSSH_10.0p2 Debian-7, OpenSSL 3.5.4 30 Sep 2025
```

La fin du fichier "Clé Privée"

Dans une configuration SSH classique, votre sécurité repose sur un fichier (id_rsa) posé sur votre disque dur. Si un malware vole ce fichier, il vole votre identité. Avec le Flipper Zero (mode U2F), la clé privée n'existe **pas** sur le PC.

- **Sur le PC :** Il ne reste qu'un "Stub" (un squelette/pointeur) qui dit au système SSH : *"Pour signer, demande au périphérique USB"*.
- **Sur le Flipper :** La signature cryptographique se fait à l'intérieur de la puce sécurisée. La clé privée ne quitte **jamais** le Flipper.

Le Mécanisme de Dérivation (Non-Résident)

Contrairement à un trousseau de clés classique, le Flipper ne stocke pas chaque clé privée générée.

- **Secret Maître :** Le dispositif possède un secret unique et immuable (*Master Secret*), sécurisé au niveau matériel.
- **Génération Dynamique :** Lors de la création d'une clé (ssh-keygen), le système combine ce Secret Maître avec le contexte (domaine/application) et un aléa pour dériver une clé privée unique.
- **Le "Key Handle" :** Le résultat n'est pas sauvegardé dans le Flipper. Il est chiffré et exporté vers l'ordinateur hôte sous la forme d'un fichier "stub" (ex: id_ecdsa_sk). Ce fichier, appelé *Key Handle*, est inexploitable sans le Flipper physique.

Processus d'Authentification

Lors de la connexion, l'hôte transmet le *Key Handle* au Flipper. Ce dernier utilise son Secret Maître pour déchiffrer le *Handle*, reconstruire la clé privée originale temporairement, signer la requête, puis effacer toute trace en mémoire volatile.

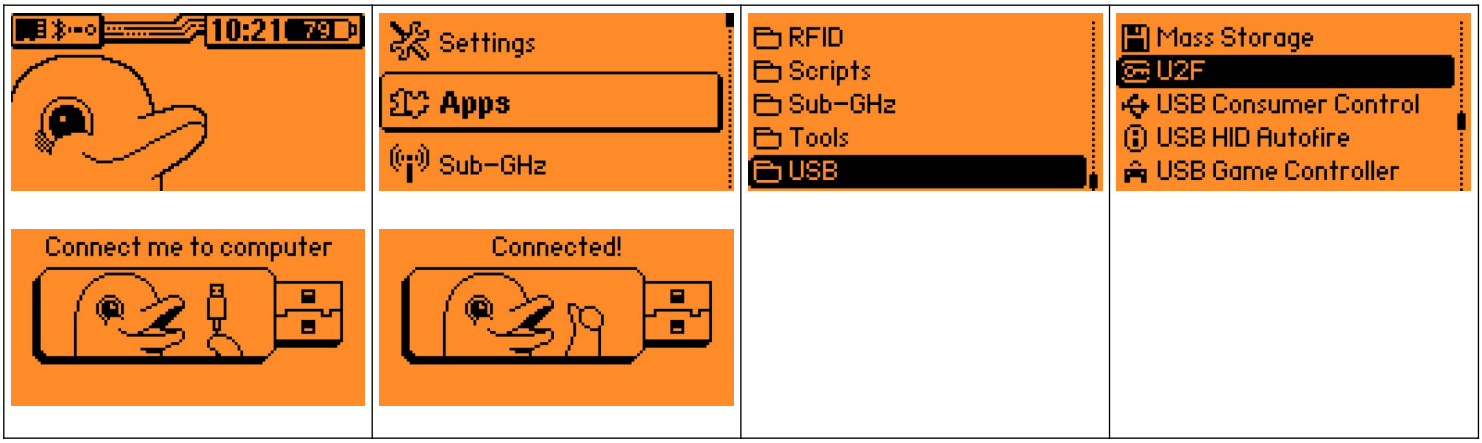
Conséquence : Ce processus permet de gérer un nombre illimité de serveurs avec un seul dispositif, sans saturation mémoire.

Exception : Les Clés Résidentes (Resident Keys)

L'option -O resident force le stockage persistant de la clé privée et des métadonnées directement dans la mémoire du Flipper (pour permettre l'authentification sans fichier *stub* sur l'hôte).

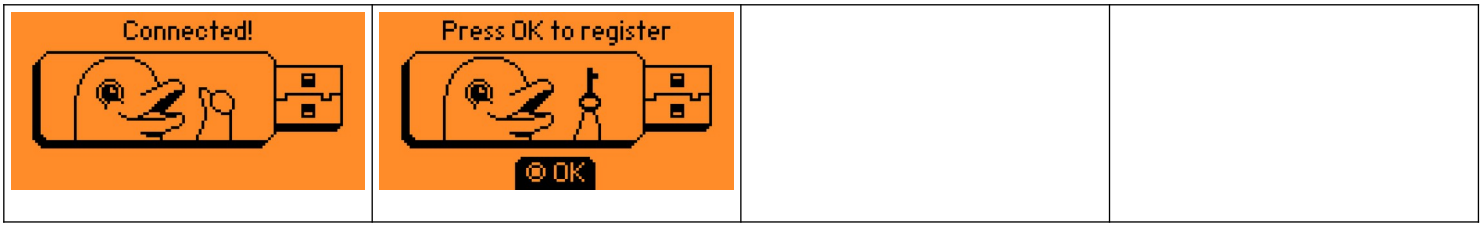
- **Contrainte :** Dans ce mode spécifique, la capacité est limitée par le stockage sécurisé du matériel (généralement entre 20 et 50 emplacements selon le firmware), contrairement au mode standard illimité.

GÉNÉRATION DE LA CLÉ (BUILD)



Nous utilisons l'algorithme ecdsa-sk car le Flipper (en émulation U2F standard) ne supporte pas encore nativement le récent ed25519-sk.

```
ssh-keygen -t ecdsa-sk -a 100 -f ~/.ssh/id_ecdsa_sk_lab-dev-01 -C "flipper0_lab-dev-01"
```




```
Generating public/private ecdsa-sk key pair.
You may need to touch your authenticator to authorize key generation.
Enter passphrase for "/home/user/.ssh/id_ecdsa_sk_atlas" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/user/.ssh/id_ecdsa_sk_ lab-dev-01
Your public key has been saved in /home/user/.ssh/id_ecdsa_sk_lab-dev-01.pub
The key fingerprint is:
SHA256:zJmPJ4SmZ0fUAZ/hnz2AEJtAnYgyBFC6DfX1oH4GDec bb8_atlas
The key's randomart image is:
+-[ECDSA-SK 256]--+
|*ooo+o*o..o      |
|oo...Oo* = +     |
|oo  + E+ *       |
| + . . = + =     |
|. . . = S o o    |
|     = o o       |
|    . o + o      |
|     o . o       |
|                 |
+----[SHA256]-----+
```

DÉPLOIEMENT & UTILISATION (RUN)

Propagation de la Clé Publique

On envoie la serrure sur le serveur lab-dev-01 (remplacez les variables par vos infos réelles) :

```
ssh-copy-id -i .ssh/id_ecdsa_sk_lab-dev-01.pub user@lab-dev-01
```

```
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: ".ssh/id_ecdsa_sk_lab-dev-01.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already
installed
Enter passphrase for key '/home/user/.ssh/lab-dev-01':
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new
keys

Enter passphrase for key '/home/seed/.ssh/lab-dev-01':

Number of key(s) added: 1

Now try logging into the machine, with: "ssh -i .ssh/id_ecdsa_sk_lab-dev-01 'user@lab-dev-01'"
and check to make sure that only the key(s) you wanted were added.
```

Simplification de la Connexion (Best Practice)

Pour éviter de taper -i ... à chaque connexion, nous déclarons la clé dans le registre client.

```
nano .ssh/config
```

```
Host lab-dev-01
    HostName IP_DU_SERVER
    User VOTRE_USER
    IdentityFile ~/.ssh/id_ecdsa_sk_lab-dev-01
```

Le Rituel de Connexion

Commande :

```
ssh lab-dev-01
```

Challenge :

```
Confirm user presence for key ECDSA-SK SHA256:zJmPJ4SmZ0fUAZ/hnz2AEJtAnYgyBFC6DfX1oH4GDec
```

Action Physique : Le Flipper vibre/clignote. **Appuyez sur le bouton.**

Résultat : Accès accordé.

```
User presence confirmed
```