Protecting SSH authentication with TPM 2.0

Nicolas IOOSS

SSTIC 2021

Utiliser un TPM 2.0 sur Linux, c'est simple !

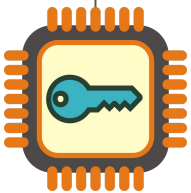
Ledger

Client SSH + tpm2-pkcs11

Serveur SSH

Authentification :
Mot de passe,
Fichier de clé SSH,
...

Commandes, fichiers...



TPM (Trusted Platform Module)

C'est simple, gratuit et plus sécurisé
qu'un MDP ou qu'un fichier !

Sources: https://fr.wikipedia.org/wiki/Fichier:Tux.svg,
https://pixabay.com/vectors/security-board-chip-computer-152690/,
https://fr.wikipedia.org/wiki/OpenSSH#/media/Fichier:Openssh.gif

Utiliser un TPM 2.0 sur Linux, c'est simple !

Ledger

```
# Pour Arch Linux :
sudo pacman -S tpm2-pkcs11

# Pour Debian 11 et Ubuntu 21.04 :
sudo apt install libtpm2-pkcs11-tools

tpm2_ptool init
tpm2_ptool addtoken --pid=1 --label=ssh --userpin=XXXX --sopin=YYYY
tpm2_ptool addkey --label=ssh --userpin=XXXX --algorithm=ecc256

ssh-keygen -D /usr/lib/pkcs11/libtpm2_pkcs11.so
ssh -I /usr/lib/pkcs11/libtpm2_pkcs11.so user@server
```

The END !

Merci

Utiliser un TPM 2.0, c’est simple !

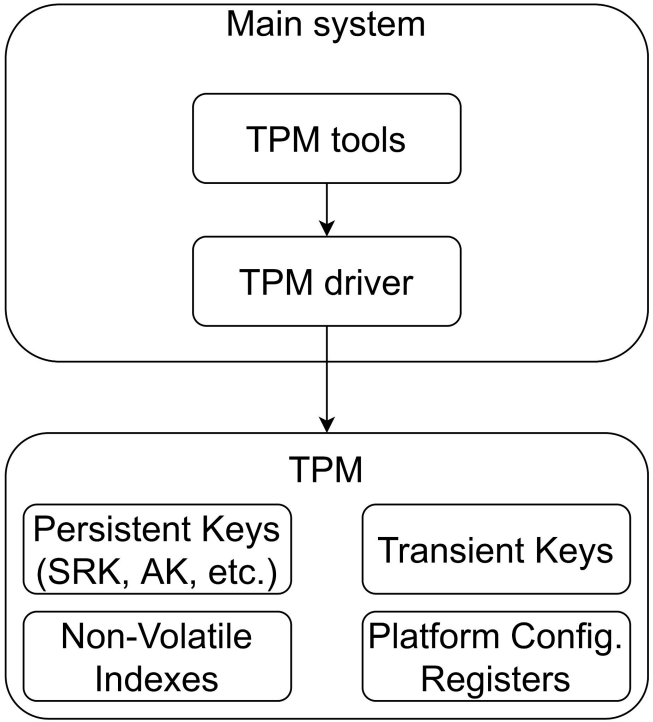
Ledger

- Comment sont enregistrées et utilisées les clés ?
- Est-il possible de les exporter ?
- Si je réinstalle mon système, est-ce que les clés sont perdues ?
- Si je mets à jour mon BIOS, est-ce que les clés sont perdues ?

Plan

1. Environnement d’expérimentation
2. Comment la clé SSH est-elle enregistrée dans le TPM ?
3. Comment le code PIN protège la clé ?

1. Environnement d’expérimentation / Qu’est ce qu’un TPM ?



1. Environnement d'expérimentation / TPM simulé

Ledger

Pour tester sans risque de « casser » un TPM : utiliser un simulateur de TPM !

- https://github.com/stefanberger/swtpm : `swtpm` (compatible QEMU et module `tpm_vtpm_proxy`)
- https://github.com/kgoldman/ibmswtpm2 : `tpm_server`

Exemple dans un conteneur Arch Linux (Podman, Docker, etc.) :

```
pacman -Syu swtpm tpm2-abrmd tpm2-tools
swtpm socket --tpm2 --daemon --server port=2321 --ctrl type=tcp,port=2322 \
   --flags not-need-init --tpmstate dir=/tmp --log file=/tmp/swtpm.log,level=5
mkdir -p /run/dbus && dbus-daemon --system --fork
tpm2-abrmd --allow-root --tcti swtpm:host=127.0.0.1,port=2321 &
export TPM2TOOLS_TCTI=tabrmd:bus_type=system
```

1. Environnement d’expérimentation / test

Ledger

Est-ce que le TPM simulé fonctionne ?

```
tpm2_getcap properties-fixed
tpm2_pcrread
tpm2_readpublic -c 0x81000000
```

Avec https://github.com/fishilico/home-files/blob/master/bin/tpm-show :

```
tpm-show --port 2321
```

Plan

1. Environnement d’expérimentation
2. Comment la clé SSH est-elle enregistrée dans le TPM ?
3. Comment le code PIN protège la clé ?

2. Comment la clé SSH est-elle enregistrée dans le TPM ?

Ledger

Deux possibilités :

1. « Comme une carte à puce » : le secret (la clé privée) ne quitte jamais le TPM.
   Seul le TPM peut utiliser le secret, par exemple pour signer des messages.
2. « Comme BitLocker » : un secret est déchiffré par le TPM et transmis à l'application (« *unseal* »).
   L'application peut donc utiliser le secret sans nécessiter le TPM.

Dans le code de tpm2-pkcs11 :

- Les deux semblent utilisées en même temps !
- Les clés privées sont chiffrées et enregistrées hors du TPM.

2. Comment la clé SSH est-elle enregistrée dans le TPM ?

Ledger

Exploration des données de tpm2-pkcs11

```
$ sqlite3 "$HOME/.tpm2_pkcs11/tpm2_pkcs11.sqlite3"
sqlite> .tables
pobjects     schema       sealobjects  tobjects     tokens
```

`tobjects` : « *transient objects* », contient les attributs d'objets PKCS#11

Deux objets :

- Une clé publique
- Une clé privée

2. Comment la clé SSH est-elle enregistrée dans le TPM ?

Ledger

Dans les attributs PKCS#11 : des champs standards, et 3 spécifiques :

- `CKA_TPM2_OBJAUTH_ENC`
- `CKA_TPM2_PUB_BLOB`: clé publique, structure `TPM2B_PUBLIC` (spécification TPM 2.0)
- `CKA_TPM2_PRIV_BLOB`: structure `TPM2B_PRIVATE`, « donnée chiffrée par le TPM »

2. Comment la clé SSH est-elle enregistrée dans le TPM ?

```
struct TPM2B_PUBLIC {
  size = 0x0056,
  publicArea = {                // struct TPMT_PUBLIC
    type = 0x0023,              // = TPM_ALG_ECC
    objectAttributes = 0x00060072,
...
    parameters.eccDetail = { curveID = 0x0003, ... }, // = TPM_ECC_NIST_P256
    unique.ecc = {              // struct TPMS_ECC_POINT
      x = {                     // struct TPM2B_ECC_PARAMETER
        size = 0x0020,
        bytes = "3eef05ada9dc42f69ffca066adfc374ec94aaba63bfa9383c2a563d847f31ac2"
      },
      y = {
        size = 0x0020,
        bytes = "50702adc8e1081d1b633a1e1d6278b4613ba20cf5fd8af0b8c3c8b4a765b9387"
      }
    }
  }
}
```

2. Comment la clé SSH est-elle enregistrée dans le TPM ?

```
struct TPM2B_PRIVATE {
  size = 0x007e,
  buffer = {
    integrity = {             // struct TPM2B_DIGEST
      size = 0x0020,
      bytes = "93b2e33a7ff39879229e35afeb86ec61bca0aaee057c0d56bee354bc41cc01f5"
    },
    iv = {
      size = 0x0010,
      bytes = "627e422444e01671fe6b2e3a771634d6"
    },
    encrypted =               // 74 octets (= 0x4a)
      "4d64599bc3129fb57f102bb89244e6d7c6c029a9a53b27bddbb0ba5b5fa0497c"
      "3286364b50fce3757615c895de4fce053c4793a4b39b35007fb7d2a29557b9b3"
      "18b15ecbd4f7c70908a8"
  }
}
```

2. Comment la clé SSH est-elle enregistrée dans le TPM ?

Ledger

Comment est chiffrée la clé privée (qui est dans l'attribut `CKA_TPM2_PRIV_BLOB`de la clé SSH générée) ?

Pour charger la clé dans le TPM, cette commande fonctionne :

```
$ tpm2_load -c /tmp/context -C 0x81000000 -u pub_blob -r priv_blob
name: 000bcac322c64b1a31d7806bc84570090949f898cea8c2c9a258761659dfb1de713d
```

`0x81000000`: Identifiant (*Handle*) de la « *Storage Root Key* » (SRK)

Réponse floue : « La clé privée est chiffrée avec la SRK du TPM »

Mais… la SRK est une clé RSA !

2. Comment la clé SSH est-elle enregistrée dans le TPM ?

Ledger

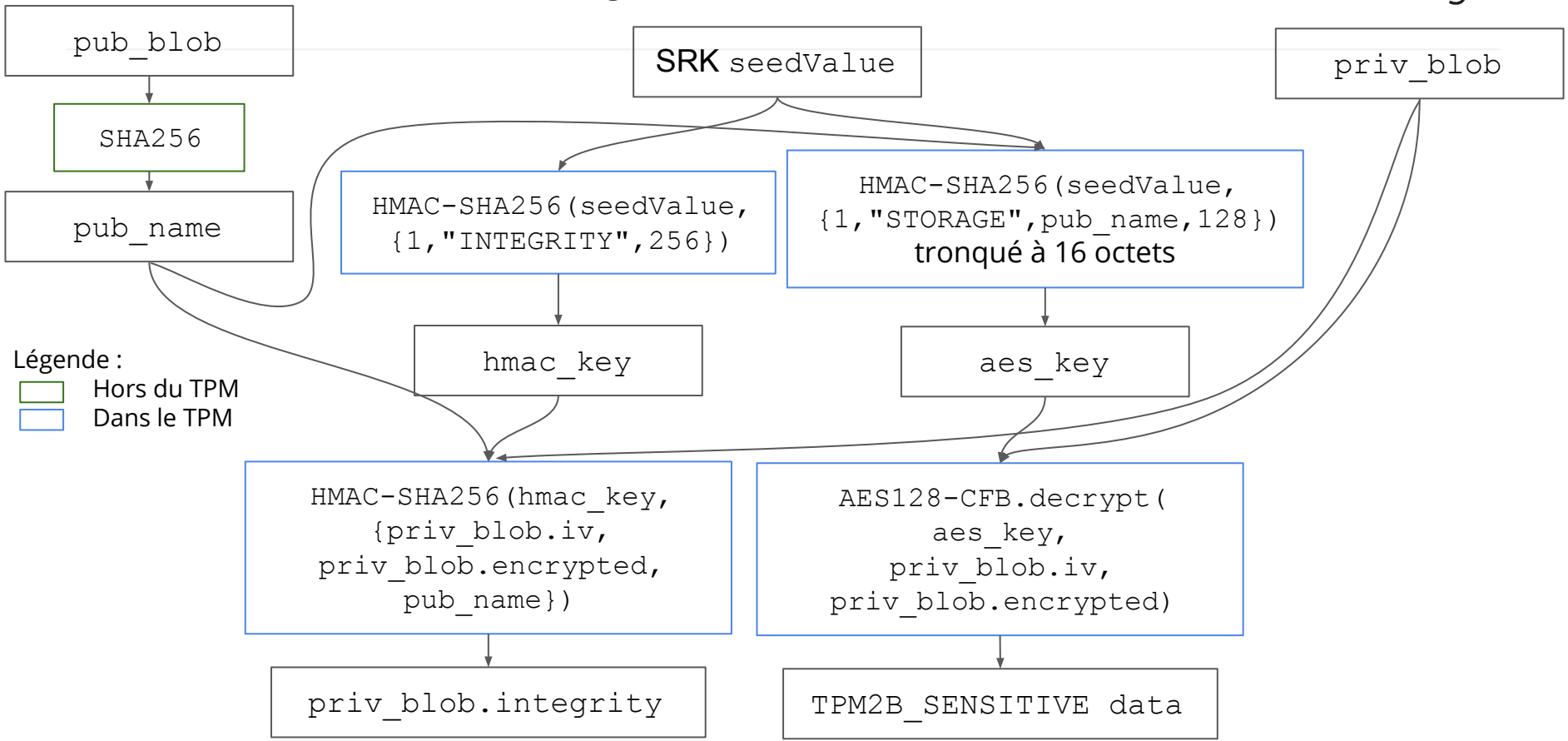
Mais… la SRK est une clé RSA… et aussi une clé AES !

Avec l'émulateur de TPM swtpm, elle est enregistrée dans `tpm2-00.permall` dans un `OBJECT` qui contient une structure `TPMT_PUBLIC` et :

```
typedef struct {
    TPMI_ALG_PUBLIC             sensitiveType;
    TPM2B_AUTH                  authValue;
    TPM2B_DIGEST                seedValue; // donnée secrète
    TPMU_SENSITIVE_COMPOSITE    sensitive; // clé privée
} TPMT_SENSITIVE;
```

(NB. La seedValue n'est pas la *Storage Primary Seed* (SPS) du TPM mais y est liée quand la SRK est dans la *Owner Hierarchy*…)

2. Comment la clé SSH est-elle enregistrée dans le TPM ?



(NB. Algorithmes par défaut, qui dépendent de champs configurables de la SRK)

2. Comment la clé SSH est-elle enregistrée dans le TPM ?

Ledger

Données déchiffrées de l'attribut `CKA_TPM2_PRIV_BLOB` en utilisant la `seedValue` de la SRK

```
struct TPM2B_SENSITIVE {
  size = 0x0048,
  sensitiveArea = {
    sensitiveType = 0x0023, // = TPM_ALG_ECC
    authValue = {      →32 caractères "06412b67afc87c07eba26c4e01ae6b50"
      size = 0x0020,
      buffer = "3036343132623637616663383763303765626132366334653031616536623530"
    },
    seedValue = { size = 0x0000 },
    sensitive.ecc = { →clé privée (scalaire de 256 bits pour la courbe NIST P-256)
      size = 0x0020,
      buffer = "e136a90d627a7b2ea404ed671a7717cb04b13f54f9df478ff54ced6fd3275048"
    }
  }
}
```

2. Comment la clé SSH est-elle enregistrée dans le TPM ?

Ledger

Bilan provisoire :

- La clé SSH générée par `tpm2_ptool` n'est pas persistante dans le TPM
- La clé privée a été chiffrée (en AES-CFB) avec la *Storage Root Key* du TPM
- La base de données de tpm2-pkcs11 contient la clé privée chiffrée

Avec un vrai TPM, le système (OpenSSH, Linux…) ne peut pas connaître la clé privée utilisée.
(sauf si une vulnérabilité dans le TPM permet de compromettre la seedValue de la SRK)

Mais à quoi sert le PIN ? Est-ce que la clé est utilisable directement, avec `tpm2_load` ?

```
$ tpm2_load -c /tmp/context -C 0x81000000 -u pub_blob -r priv_blob
$ echo hello | tpm2_sign -c /tmp/context -g sha256 -s ecdsa -o signature.out
WARNING:esys:src/tss2-esys/api/Esys_Sign.c:311:Esys_Sign_Finish() Received TPM Error
ERROR:esys:src/tss2-esys/api/Esys_Sign.c:105:Esys_Sign() Esys Finish ErrorCode (0x0000098e)
ERROR: Eys_Sign(0x98E) - tpm:session(1):
the authorization HMAC check failed and DA counter incremented
```

(DA signifie *Dictionary Attack*)

```
ERROR: Unable to run tpm2_sign
```

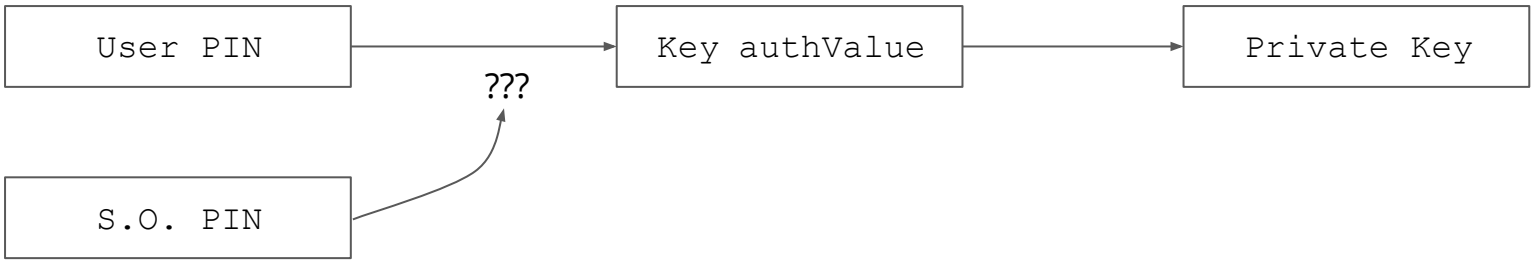
Plan

1. Environnement d’expérimentation
2. Comment la clé SSH est-elle enregistrée dans le TPM ?
3. Comment le code PIN protège la clé ?

3. Comment le code PIN protège la clé ?

```
tpm2_ptool addtoken --pid=1 --label=ssh --userpin=XXXX --sopin=YYYY
```

SOPIN = Security Officer Personal Identification Number (permet de réinitialiser le code PIN)

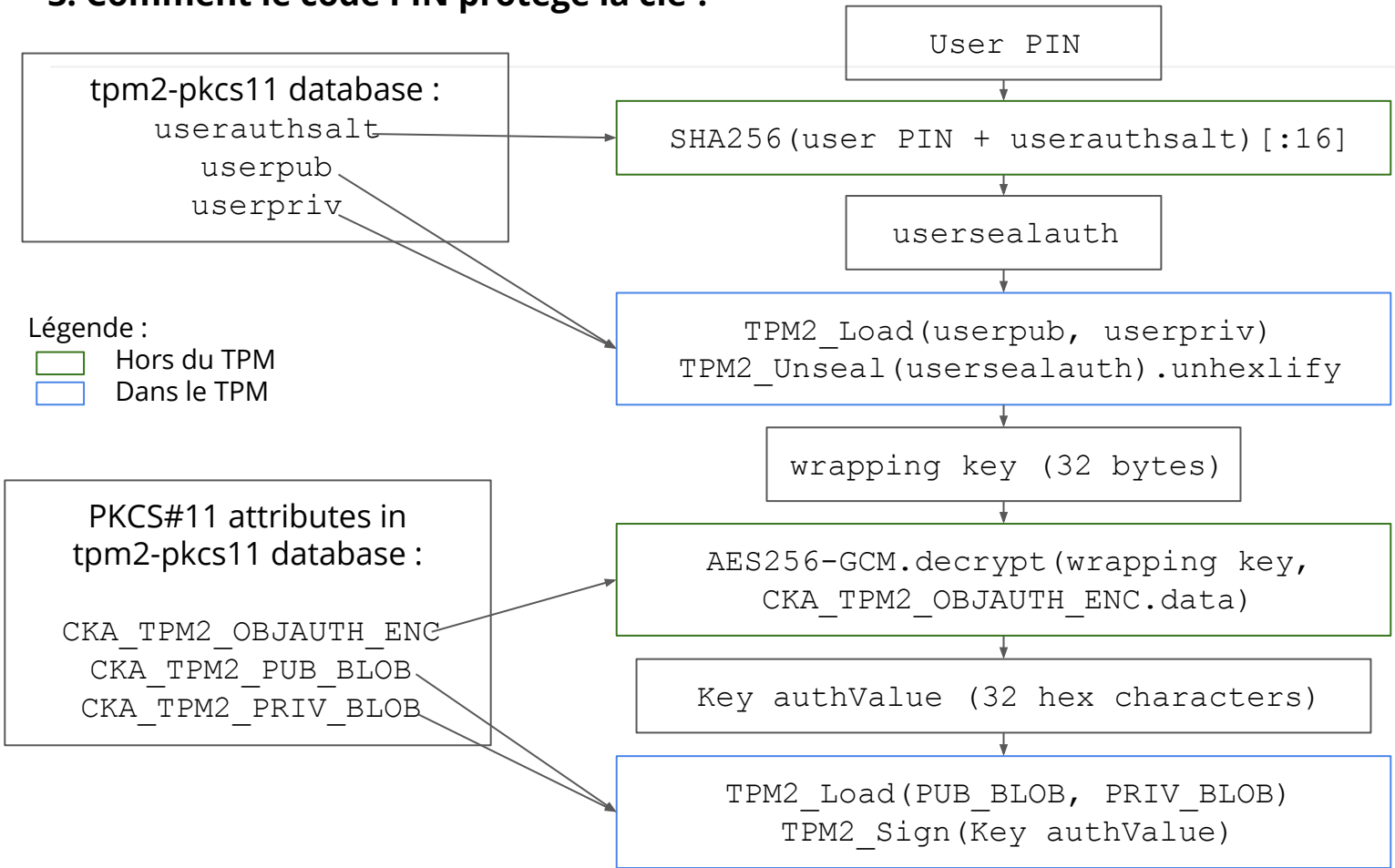


3. Comment le code PIN protège la clé ?

```
$ sqlite3 "$HOME/.tpm2_pkcs11/tpm2_pkcs11.sqlite3"
sqlite> .dump sealobjects
PRAGMA foreign_keys=OFF;
BEGIN TRANSACTION;
CREATE TABLE sealobjects(
    id INTEGER PRIMARY KEY,
    tokid INTEGER NOT NULL,
    userpub BLOB,                => TPM2B_PUBLIC (type=TPM_ALG_KEYEDHASH)
    userpriv BLOB,               => TPM2B_PRIVATE (chiffrée avec SRK seedValue)
    userauthsalt TEXT,
    sopub BLOB NOT NULL,         => TPM2B_PUBLIC (type=TPM_ALG_KEYEDHASH)
    sopriv BLOB NOT NULL,        => TPM2B_PRIVATE (chiffrée avec SRK seedValue)
    soauthsalt TEXT NOT NULL,
    FOREIGN KEY (tokid) REFERENCES tokens(id) ON DELETE CASCADE
);
```

Stockage de clé symétrique ???

3. Comment le code PIN protège la clé ?



Conclusion

Ledger

- La clé SSH générée par `tpm2_ptool` n'est pas persistante dans le TPM
- La clé privée a été chiffrée (en AES-CFB) avec la *Storage Root Key* du TPM
- La base de données de tpm2-pkcs11 contient la clé privée chiffrée

Et :

- Les codes PIN sont utilisés pour *unseal* une clé de chiffrement pour obtenir `authValue`
- Pour conserver ses clés lors de la réinstallation d'un système, il suffit de sauvegarder `"$HOME/.tpm2_pkcs11/tpm2_pkcs11.sqlite3"`, sauf si la SRK du TPM est réinitialisée.

Avec un vrai TPM, le système (OpenSSH, Linux…) ne peut pas connaître la clé privée utilisée.

Questions ?