

# PROTECTING AUTHENTICATION KEYS WITH A TPM 2.0

Nicolas Iooss

Barbhack 2021



## Nicolas Iooss

- ▶ Security Engineer at Ledger (Donjon)
- ▶ SELinux Developer/Maintainer (#2 contributor<sup>1</sup> of the userspace part)
- ▶ Trainer at SecureSphere by EPITA (teaching secure development for companies)

<https://github.com/fishilico>, <https://twitter.com/looNag>

---

<sup>1</sup><https://github.com/SELinuxProject/selinux/graphs/contributors>



SSH client



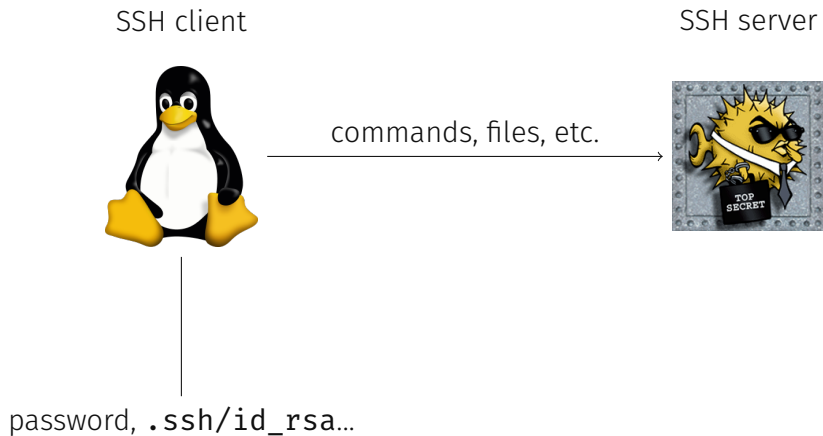
commands, files, etc.



SSH server



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



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

SSH client



commands, files, etc.



SSH server



USB



Ledger Nano S/X

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

SSH client



commands, files, etc.



SSH server



USB



Ledger Nano S/X (60-120 €)

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

SSH client + tpm2-pkcs11

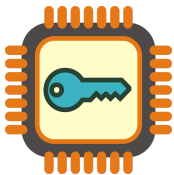
SSH server



commands, files, etc.



Internal bus



TPM (Trusted Platform Module), integrated with laptops

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

```
sudo apt install libtpm2-pkcs11-1 libtpm2-pkcs11-tools
```

```
tpm2_ptool init
```

```
tpm2_ptool addtoken --pid=1 --label=ssh --userpin=MyPassword --sopin=MyRecoveryPassword
```

```
tpm2_ptool addkey --label=ssh --userpin=MyPassword --algorithm=ecc256
```

```
ssh-keygen -D /usr/lib/x86_64-linux-gnu/libtpm2_pkcs11.so.1
```

```
ssh -I /usr/lib/x86_64-linux-gnu/libtpm2_pkcs11.so.1 server
```

```
# For Arch Linux:
```

```
# sudo pacman -S tpm2-pkcs11
```

```
# and use /usr/lib/pkcs11/libtpm2_pkcs11.so
```





- ▶ How are the private keys stored?
- ▶ Can malware steal the private keys?
- ▶ How can the keys be backed up?



- ▶ How are the private keys stored?
- ▶ Can malware steal the private keys?
- ▶ How can the keys be backed up?

⇒ [https://www.sstic.org/2021/presentation/protecting\\_ssh\\_authentication\\_with\\_tpm\\_20/](https://www.sstic.org/2021/presentation/protecting_ssh_authentication_with_tpm_20/)

- ▶ SQLite database in `$HOME/.tpm2_pkcs11/tpm2_pkcs11.sqlite3`
- ▶ SSH key encrypted with a *Storage Root Key* (SRK), only known to the TPM.



- ▶ How are the private keys stored?
- ▶ Can malware steal the private keys?
- ▶ How can the keys be backed up?

⇒ [https://www.sstic.org/2021/presentation/protecting\\_ssh\\_authentication\\_with\\_tpm\\_20/](https://www.sstic.org/2021/presentation/protecting_ssh_authentication_with_tpm_20/)

- ▶ SQLite database in `$HOME/.tpm2_pkcs11/tpm2_pkcs11.sqlite3`
- ▶ SSH key encrypted with a *Storage Root Key* (SRK), only known to the TPM.

Today:

- ▶ How to import existing RSA keys from file?
- ▶ How to use the PKCS#11 interface in other applications? For example for VPN authentication?



1. Simulating a TPM (testing setup)
2. Importing an existing private key
3. OpenVPN Authentication



# 1. SIMULATING A TPM: TPM 2.0 ARCHITECTURE

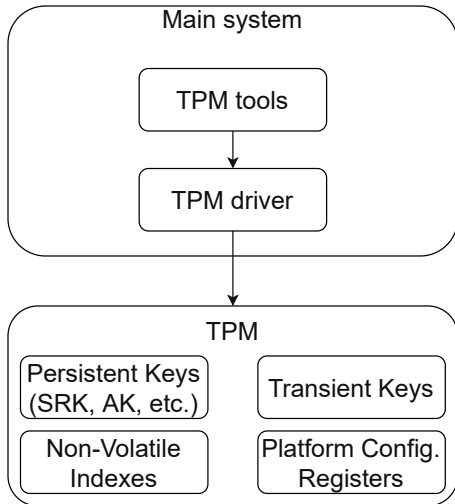


Figure 1: Components of a TPM 2.0 chip

# 1. SIMULATING A TPM 2.0

TPM simulators:

- ▶ `swtpm` from <https://github.com/stefanberger/swtpm> (compatible with QEMU)
- ▶ `tpm_server` from <https://github.com/kgoldman/ibmswtpm2>

Example on Arch Linux:

```
pacman -Sy 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. SIMULATING A TPM: IS IT WORKING?

```
# Read properties (manufacturer, firmware version, etc.)
tpm2_getcap properties-fixed

# Read the Platform Configuration Registers
tpm2_pcrread

# Read the Storage Root Key (SRK) attributes and public key
tpm2_readpublic -c 0x81000001

# Initialize a tpm2-pkcs11 storage
tpm2_ptool init
tpm2_ptool addtoken --pid=1 --label=ssh --userpin=MyPassword --sopin=MyRecoveryPassword
```



1. Simulating a TPM (testing setup)
2. Importing an existing private key
3. OpenVPN Authentication





## 2. IMPORTING AN EXISTING PRIVATE KEY

What possibilities?

- ▶ TPM 2.0 Library<sup>2</sup> Part 3: Commands: TPM2\_Import
- ▶ tpm2-tools programs: tpm2\_import

```
$ tpm2_import --help
```

```
Usage: tpm2_import [<options>]
```

```
Where <options> are:
```

```
[ -P | --parent-auth=<value>] [ -p | --key-auth=<value>]
[ -G | --key-algorithm=<value>] [ -i | --input=<value>]
[ -C | --parent-context=<value>] [ -U | --parent-public=<value>]
[ -r | --private=<value>] [ -u | --public=<value>]
[ -a | --attributes=<value>] [ -g | --hash-algorithm=<value>]
[ -s | --seed=<value>] [ -L | --policy=<value>]
[ -k | --encryption-key=<value>] [ --passin=<value>] [ --cphash=<value>]
```

---

<sup>2</sup><https://trustedcomputinggroup.org/resource/tpm-library-specification/>



## 2. IMPORTING AN EXISTING PRIVATE KEY: SIMPLE EXAMPLE

- ▶ Using TPM2\_Import command:

```
# Generate a private RSA-2048 key with OpenSSL
openssl genrsa -out rsa.pem -aes-256-ctr -passout pass:MySecret 2048

# Import it in the TPM, producing pub.blob and priv.blob
tpm2_import -C 0x81000001 -G rsa -i rsa.pem -u pub.blob -r priv.blob

tpm2_print -t TPM2B_PUBLIC pub.blob
```

- ▶ Using tpm2-pkcs11:

```
tpm2_ptool import --label=ssh --privkey=rsa.pem --userpin=MyPassword
```



## 2. IMPORTING AN EXISTING PRIVATE KEY: WITH SSH

Problem: OpenSSH private keys are not formatted in (OpenSSL) DER or PEM format!

```
ssh-keygen -t rsa -b 2048
```

```
cat ~/.ssh/id_rsa
```

```
# => starts with -----BEGIN OPENSSH PRIVATE KEY-----
```

```
tpm2_import -C 0x81000001 -G rsa -i ~/.ssh/id_rsa -u pub.blob -r priv.blob
```

```
# => ERROR: Reading PEM file "/home/user/.ssh/id_rsa" failed
```

How to convert from OpenSSH to DER/PEM format?



## 2. IMPORTING AN EXISTING PRIVATE KEY: WITH SSH

Problem: OpenSSH private keys are not formatted in (OpenSSL) DER or PEM format!

```
ssh-keygen -t rsa -b 2048
```

```
cat ~/.ssh/id_rsa
```

```
# => starts with -----BEGIN OPENSSSH PRIVATE KEY-----
```

```
tpm2_import -C 0x81000001 -G rsa -i ~/.ssh/id_rsa -u pub.blob -r priv.blob
```

```
# => ERROR: Reading PEM file "/home/user/.ssh/id_rsa" failed
```

How to convert from OpenSSH to DER/PEM format?

# Warning: "ssh-keygen -p" modifies the key file directly! Make a backup before running!

```
ssh-keygen -p -f ~/.ssh/id_rsa -m pem
```

```
tpm2_ptool import --label=ssh --privkey=$HOME/.ssh/id_rsa --userpin=MyPassword
```



## 2. IMPORTING AN EXISTING PRIVATE KEY

The screenshot shows a web browser window displaying a GitHub issue page. The browser's address bar shows the URL `github.com/tpm2-software/tpm2-pkcs11/issues/659`. The page title is "Feature request: import SSH private key using tpm2\_ptool #659". The repository name is "tpm2-software / tpm2-pkcs11". The issue is marked as "Closed" and was opened by "fishilico" on Feb 22. The issue content includes a greeting, a link to documentation, and a list of features. The right sidebar shows "Assignees", "Labels", "Projects", and "Milestone" sections, all of which are currently empty.

Feature request: import SSH private key using tpm2\_ptool - Issue #659 · tpm2-software/tpm2-pkcs11 · GitHub · Chromium

Feature request: import SSH private key using tpm2\_ptool

github.com/tpm2-software/tpm2-pkcs11/issues/659

tpm2-software / tpm2-pkcs11

Notifications Star 128 Fork 62

<> Code Issues 43 Pull requests 2 Actions Projects 1 Security Insights

Feature request: import SSH private key using tpm2\_ptool #659

New Issue

Closed fishilico opened this issue on Feb 22 · 1 comment

fishilico commented on Feb 22 · edited

Hello,

It is currently possible to use keys from tpm2-pkcs11 in OpenSSH thanks to the great documentation in <https://github.com/tpm2-software/tpm2-pkcs11/blob/1.5.0/docs/SSH.md>. More over:

- Users are able to quickly create new SSH keys using `tpm2_ptool.py addkey --algorithm=rsa2048 --label=label --userpin=myuserpin ... (or --algorithm=ecc256)`
- Users are able to quickly import RSA and EC private keys stored in PEM format using `tpm2_ptool.py import ...`

However users are not able to quickly import RSA and EC private keys stored in OpenSSH format, because OpenSSH uses a custom format to store its keys.

Assignees  
No one assigned

Labels  
None yet

Projects  
None yet

Milestone  
No milestone



## 2. IMPORTING AN EXISTING PRIVATE KEY

The screenshot shows a web browser window displaying a GitHub Pull Request (PR) page. The browser's address bar shows the URL `github.com/tpm2-software/tpm2-pkcs11/pull/681`. The page title is `tpm2_ptool import: enable importing encrypted private keys and SSH private keys #681`. The repository is `tpm2-software / tpm2-pkcs11`, which has 128 stars and 62 forks. The PR is in the `pull requests` tab, showing it is `Merged` by `williamcroberts` on May 24. The PR description states: "Hello, This Pull Request implements what was discussed in #659 : being to able to use `tpm2_ptool import` with OpenSSH keys. In order to support encrypted private keys, I added option `--passin` exactly like `tpm2_import`. This enables importing encrypted PEM and DER private keys with all supported `--passin` variants (cf. `man 1 openssl` : <https://www.openssl.org/docs/man1.1.1/man1/openssl.html#Pass-Phrase-Options>). This option is also parsed when importing encrypted OpenSSH keys. While at it, this PR also makes `--algorithm` optional, as the type of the private key (`ecc` or `rsa`)". The PR has 2 conversations, 5 commits, 16 checks, and 2 files changed, with a net change of +76 -8 lines. The right sidebar shows sections for Reviewers, Assignees, Labels, and Projects, all of which are currently empty.



## 2. IMPORTING AN EXISTING PRIVATE KEY: WITH SSH

Importing an existing SSH private key is too complex.

Solution: <https://github.com/tpm2-software/tpm2-pkcs11/pull/681>

*tpm2\_ptool import: enable importing encrypted private keys and SSH private keys*

In the next tpm2-pkcs11 release (1.7.0), this will work:

```
tpm2_ptool import --label=ssh --privkey=$HOME/.ssh/id_rsa --userpin=MyPassword
```



## 2. IMPORTING AN EXISTING PRIVATE KEY: OTHER ISSUES?

When does importing fail?

- ▶ The algorithm is not supported by the TPM (Ed25519)
- ▶ The key size is not supported by the TPM (RSA 4096)

How to check whether a TPM supports RSA 4096?

```
$ tpm2_testparms rsa4096  
ERROR: Unsupported algorithm specification  
ERROR: Unable to run tpm2_testparms
```





1. Simulating a TPM (testing setup)
2. Importing an existing private key
3. OpenVPN Authentication



### 3. OPENVPN AUTHENTICATION

Generating a client signature request from a TPM key:

```
tpm2_ptool addtoken --pid=1 --label=openvpn --userpin=MyPassword \  
  --sopin=MyRecoveryPassword  
tpm2_ptool addkey --label=openvpn --algorithm=rsa2048
```

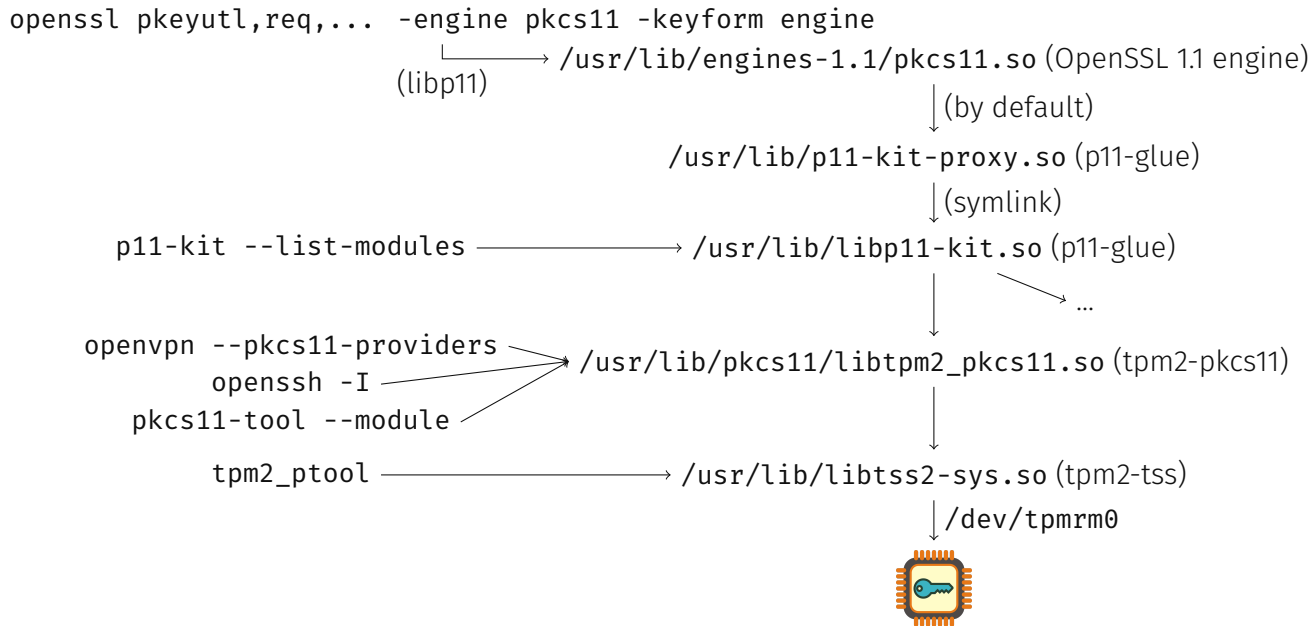
```
p11tool --list-token-urls  
# => read the line matching the token.  
# For example: pkcs11:model=SW%20%20%20TPM;manufacturer=IBM;serial=0000000000000000;  
#   token=openvpn
```

```
p11tool --login --list-all "${TOKEN}"  
# => read the URL matching the private key.  
# For example: pkcs11:model=SW%20%20%20TPM;manufacturer=IBM;serial=0000000000000000;  
#   token=openvpn;id=%64%39%64%66%37%35%66%62%31%36%63%37%31%62%65%65;type=private
```

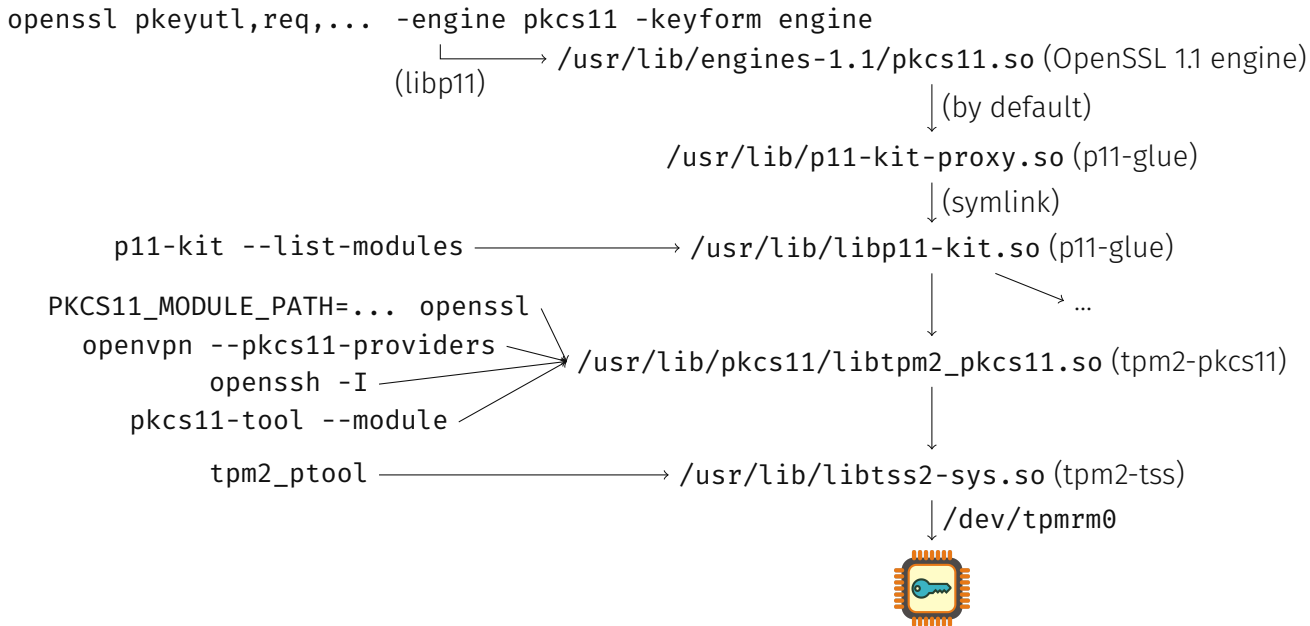
```
openssl req -new -engine pkcs11 -keyform engine -key "${PRIVATE_KEY}" -out client.csr
```



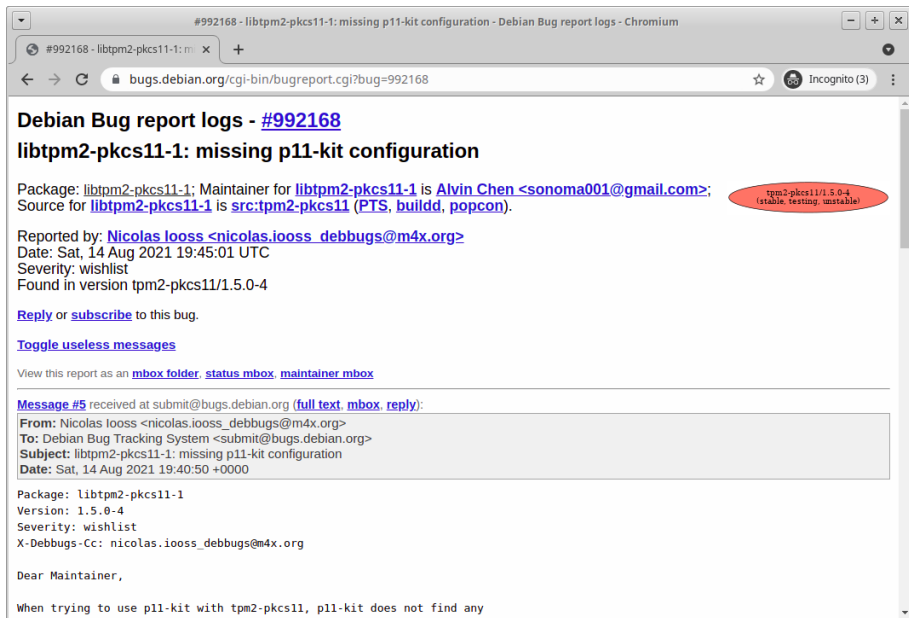
### 3. OPENVPN AUTHENTICATION (LIBRARIES ON AN ARCH LINUX SYSTEM)



### 3. OPENVPN AUTHENTICATION (LIBRARIES ON AN ARCH LINUX SYSTEM)



### 3. OPENVPN AUTHENTICATION: DEBIAN BUG #992168



The screenshot shows a Chromium browser window displaying the Debian bug report for #992168. The page title is "#992168 - libtpm2-pkcs11-1: missing p11-kit configuration - Debian Bug report logs - Chromium". The URL is "bugs.debian.org/cgi-bin/bugreport.cgi?bug=992168". The page content includes the following information:

- Debian Bug report logs - #992168**
- libtpm2-pkcs11-1: missing p11-kit configuration**
- Package: `libtpm2-pkcs11-1`; Maintainer for `libtpm2-pkcs11-1` is [Alvin Chen <sonoma001@gmail.com>](mailto:Alvin.Chen@sonoma001@gmail.com);  
Source for `libtpm2-pkcs11-1` is [src:tpm2-pkcs11](#) ([PTS](#), [buildd](#), [popcon](#)).
- Reported by: [Nicolas Iooss <nicolas.iooss\\_debbugs@m4x.org>](mailto:Nicolas.iooss@debbugs@m4x.org)
- Date: Sat, 14 Aug 2021 19:45:01 UTC
- Severity: wishlist
- Found in version `tpm2-pkcs11/1.5.0-4`
- [Reply](#) or [subscribe](#) to this bug.
- [Toggle useless messages](#)
- View this report as an [mbox folder](#), [status mbox](#), [maintainer mbox](#)
- Message #5** received at [submit@bugs.debian.org](mailto:submit@bugs.debian.org) ([full text](#), [mbox](#), [reply](#)):
- From:** Nicolas Iooss <[nicolas.iooss\\_debbugs@m4x.org](mailto:nicolas.iooss_debbugs@m4x.org)>
- To:** Debian Bug Tracking System <[submit@bugs.debian.org](mailto:submit@bugs.debian.org)>
- Subject:** libtpm2-pkcs11-1: missing p11-kit configuration
- Date:** Sat, 14 Aug 2021 19:40:50 +0000
- Package: `libtpm2-pkcs11-1`
- Version: `1.5.0-4`
- Severity: wishlist
- X-Debbugs-Cc: [nicolas.iooss\\_debbugs@m4x.org](mailto:nicolas.iooss_debbugs@m4x.org)
- Dear Maintainer,
- When trying to use `p11-kit` with `tpm2-pkcs11`, `p11-kit` does not find any

A red oval highlights the version string `tpm2-pkcs11/1.5.0-4` with the note "(stable, testing, unstable)".



### 3. OPENVPN AUTHENTICATION

Importing a signed client certificate and using it to connect:

```
tpm2_ptool listobjects --label=openvpn  
# => read "CKA_ID:"
```

```
tpm2_ptool addcert --label=openvpn --key-id=${CKA_ID} client.crt
```

```
openvpn --show-pkcs11-ids /usr/lib/pkcs11/libtpm2_pkcs11.so  
# => read "Serialized id:"
```

```
openvpn \  
  --pkcs11-providers /usr/lib/pkcs11/libtpm2_pkcs11.so \  
  --pkcs11-id "${SERIALIZED_ID}" ...
```

Documentation: <https://github.com/tpm2-software/tpm2-pkcs11/blob/1.6.0/docs/OPENVPN.md>



TPM can be used to protect authentication keys:

- ▶ for SSH,
- ▶ for VPN,
- ▶ and much more applications!

Questions?





How to migrate SSH keys with no passphrase?

Since <https://github.com/tpm2-software/tpm2-pkcs11/pull/695>

*Enable using objects with no user PIN*

In the next tpm2-pkcs11 release (1.7.0):

```
tpm2_ptool init
tpm2_ptool addtoken --pid=1 --label=ssh --userpin= --sopin=MyRecoveryPassword
tpm2_ptool addkey --label=ssh --algorithm=ecc256
```

⇒ `--userpin` becomes optional in PIN-less tokens.

