PROTECTING AUTHENTICATION KEYS WITH A TPM 2.0

Nicolas Iooss

Barbhack 2021

WHO AM I?

`Nicolas Iooss`

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

https://github.com/fishilico, https://twitter.com/IooNag

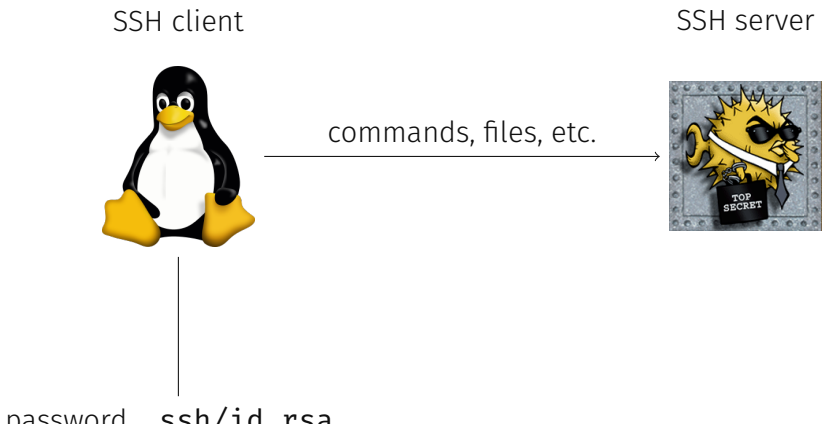
[1] https://github.com/SELinuxProject/selinux/graphs/contributors

SSH client

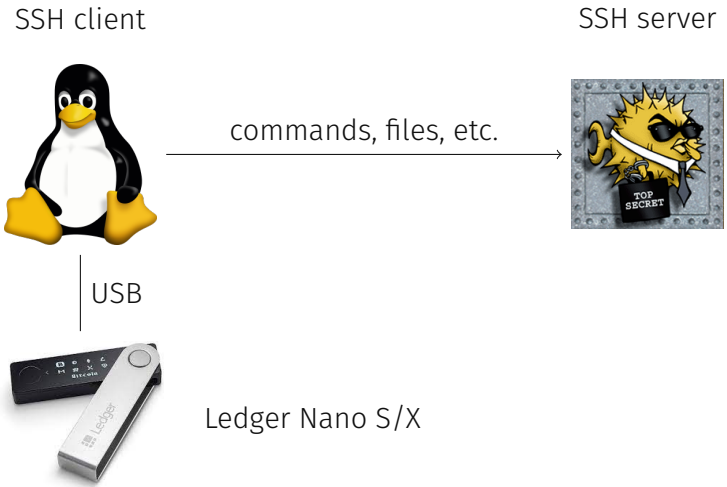
SSH server

commands, files, etc.

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/



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

SSH server

commands, files, etc.

USB

Ledger Nano S/X (60-120 €)

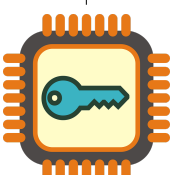
Images from: https://fr.wikipedia.org/wiki/Fichier:Tux.svg,
https://fr.wikipedia.org/wiki/OpenSSH#/media/Fichier:Openssh.gif,
https://pixabay.com/vectors/security-board-chip-computer-152690/

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/

- 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/

- 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?

AGENDA

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

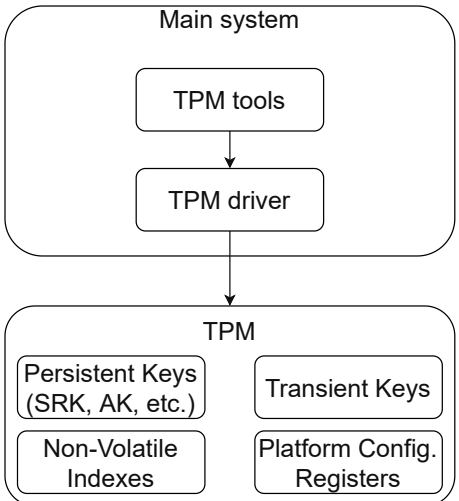


Figure 1: Components of a TPM 2.0 chip

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
```

```
# 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
```

AGENDA

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

What possibilities?

- TPM 2.0 Library[2] 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>]
```

[2] https://trustedcomputinggroup.org/resource/tpm-library-specification/

- 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
```

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?

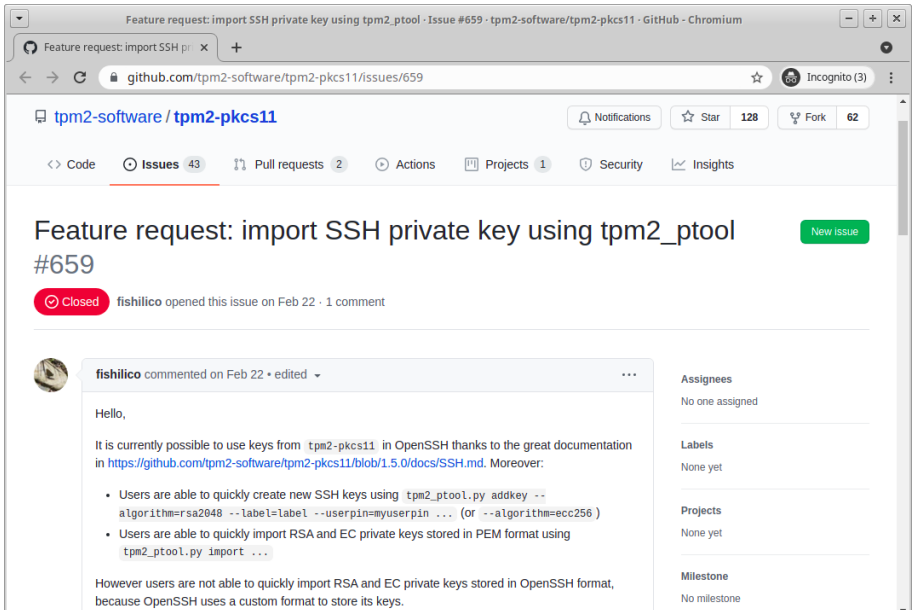
Problem: OpenSSH private keys are not formatted in (OpenSSL) DER or PEM format!

```
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
```

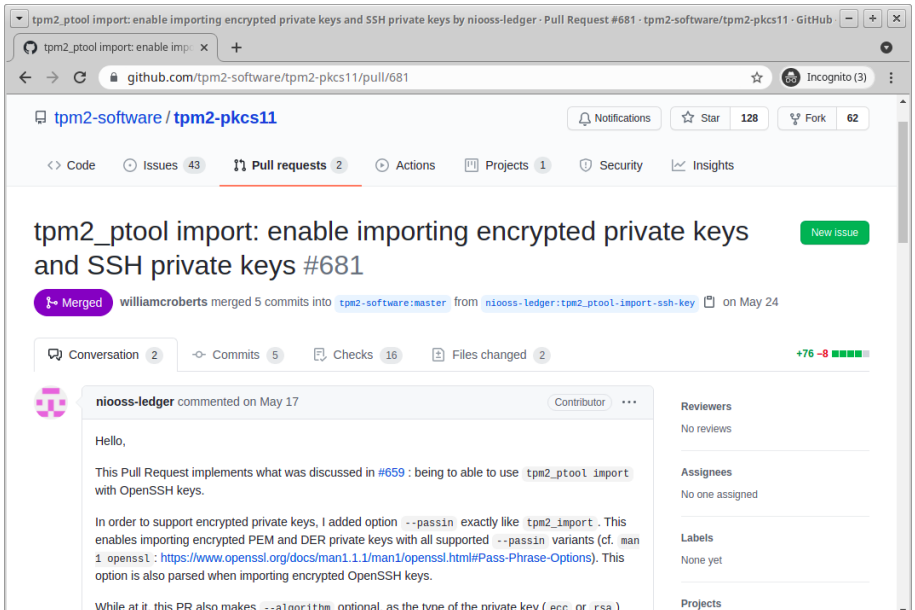
How to convert from OpenSSH to DER/PEM format?

```
# 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



2. IMPORTING AN EXISTING PRIVATE KEY



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
```

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?

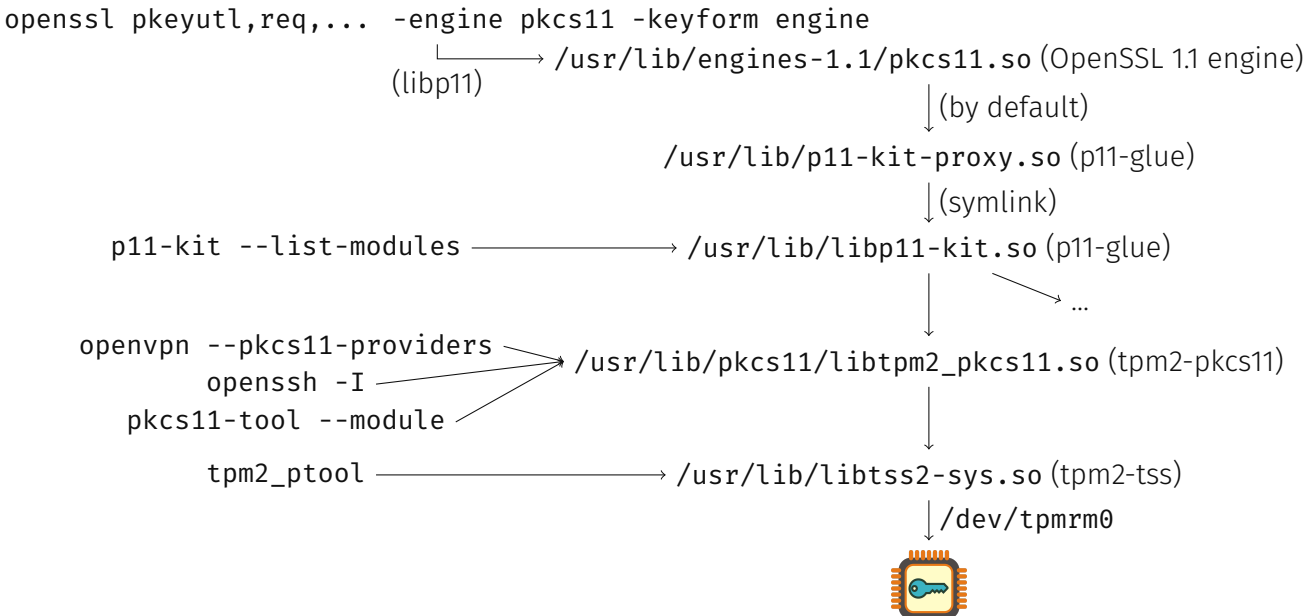
```
$ tpm2_testparms rsa4096
ERROR: Unsupported algorithm specification
ERROR: Unable to run tpm2_testparms
```

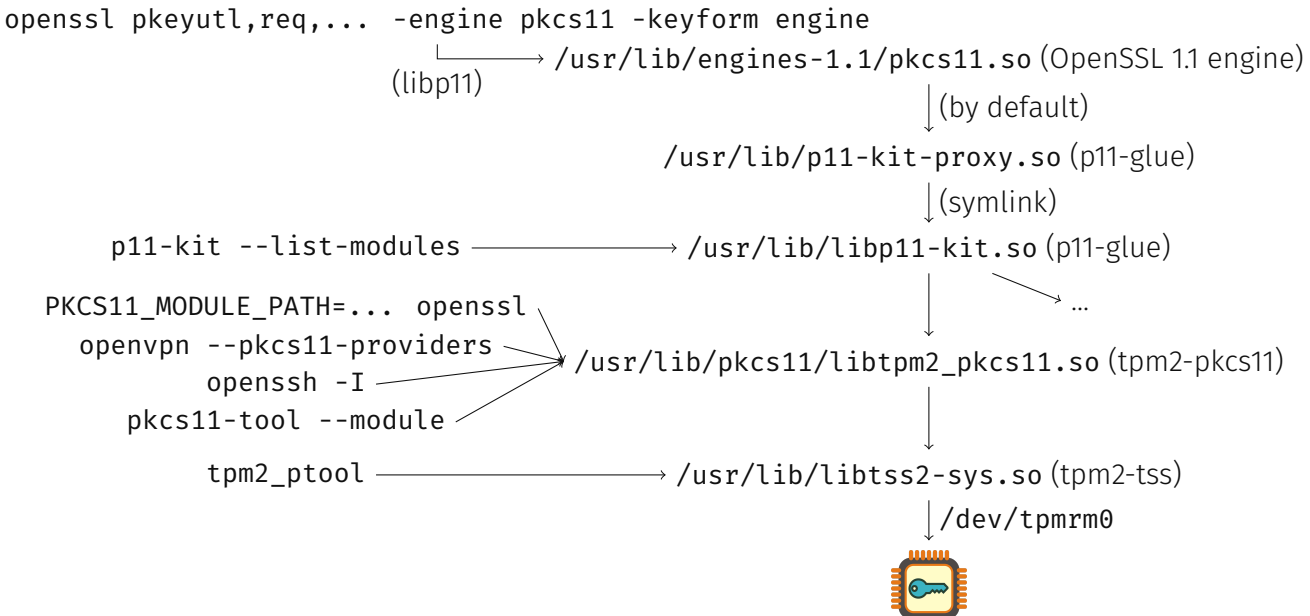
AGENDA

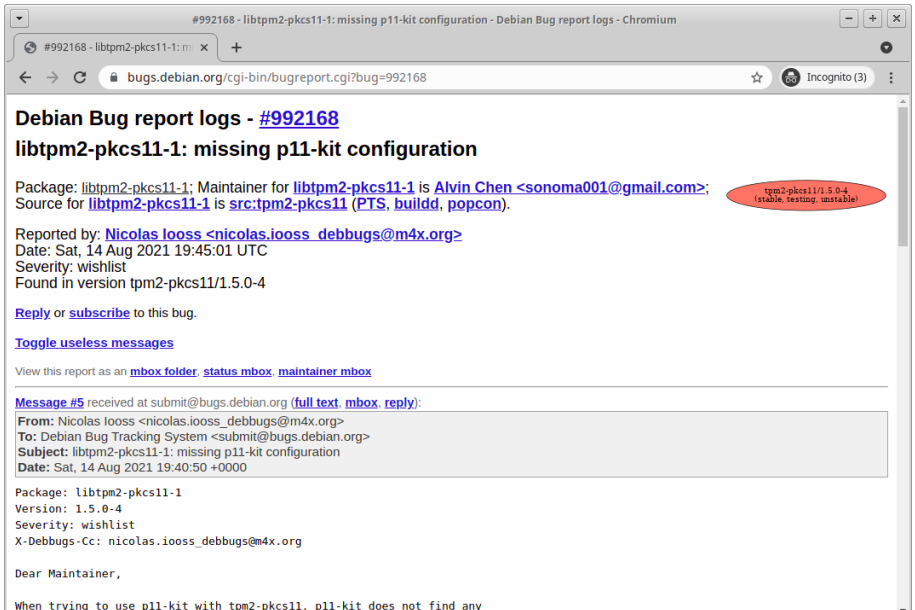
1. Simulating a TPM (testing setup)
2. Importing an existing private key
3. OpenVPN Authentication

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
```


openssl pkeyutl,req,... -engine pkcs11 -keyform engine
(libp11)
/usr/lib/engines-1.1/pkcs11.so (OpenSSL 1.1 engine)
(by default)
/usr/lib/p11-kit-proxy.so (p11-glue)
(symlink)
p11-kit --list-modules
/usr/lib/libp11-kit.so (p11-glue)
...
openvpn --pkcs11-providers
openssh -I
pkcs11-tool --module
/usr/lib/pkcs11/libtpm2_pkcs11.so (tpm2-pkcs11)
tpm2_ptool
/usr/lib/libtss2-sys.so (tpm2-tss)
/dev/tpmrm0


openssl pkeyutl,req,... -engine pkcs11 -keyform engine
(libp11)
/usr/lib/engines-1.1/pkcs11.so (OpenSSL 1.1 engine)
(by default)
/usr/lib/p11-kit-proxy.so (p11-glue)
(symlink)
p11-kit --list-modules
/usr/lib/libp11-kit.so (p11-glue)
...
PKCS11_MODULE_PATH=... openssl
openvpn --pkcs11-providers
openssh -I
pkcs11-tool --module
/usr/lib/pkcs11/libtpm2_pkcs11.so (tpm2-pkcs11)
tpm2_ptool
/usr/lib/libtss2-sys.so (tpm2-tss)
/dev/tpmrm0


#992168 - libtpm2-pkcs11-1: missing p11-kit configuration - Debian Bug report logs - Chromium
#992168 - libtpm2-pkcs11-1: m
bugs.debian.org/cgi-bin/bugreport.cgi?bug=992168
Incognito (3)
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>;
Source for libtpm2-pkcs11-1 is src:tpm2-pkcs11 (PTS, buildd, popcon).
tpm2-pkcs11/1.5.0-4 (stable, testing, unstable)
Reported by: Nicolas Iooss <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 (full text, mbox, reply):
From: Nicolas Iooss <nicolas.iooss_debbugs@m4x.org>
To: Debian Bug Tracking System <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
Dear Maintainer,
When trying to use p11-kit with tpm2-pkcs11, p11-kit does not find any

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.