Enabling TPM based system security features

Andreas Fuchs
<andreas.fuchs@sit.fraunhofer.de>

Who am I ?

- **13 year on/off TPMs**
- **Fraunhofer SIT: Trustworthy Platforms**
- **TCG-member: TPM Software Stack WG**
- **Maintainer**
  - tpm2-tss: The libraries
  - tpm2-tss-engine: The openssl engine
  - tpm2-totp: Computer-to-user attestation (mjg’s tpm-totp reimplemented for 2.0)

The hardware stack

- **Trusted Platform Module (TPM) 2.0**
  - Smartcard-like capabilities but soldered in
  - Remote Attestation capabilities
  - As separate chip (LPC, SPI, I²C)
  - In Southbridge / Firmware
  - Via TEEs/TrustZone, etc
  - Thanks to Windows-Logos in every PC
- **CPU**
  - OS, TSS 2.0, where the fun is...

The TPM Software Stack 2.0

- **Kernel exposes /dev/tpm0 with byte buffers**
- **tpm2-tss is like the mesa of TCG specs**
- **TCG specifications:**
  - TPM spec for functionality
  - TSS spec for software API
- **tpm2-tss implements the glue**
- **Then comes core module / application integration**
  - Think GDK, but OpenSSL
  - Think godot, but pkcs11
  - Think wayland, but cryptsetup

The TSS APIs

System API (sys)
- 1:1 to TPM2 cmds
- Cmd / Rsp serialization
- No file I/O
- No crypto
- No heap / malloc

Enhanced SYS (esys)
- Automate crypto for HMAC / encrypted sessions
- Dynamic TCTI loading
- Memory allocations
- No file I/O

Feature API (FAPI)
- Spec in draft form
- TBimplemented
- No custom typedefs
- JSON interfaces
- Provides Policy language
- Provides keystore

User Space

TPM Command Transmission Interface (tss2-tcti)
- Abstract command / response mechanism,
- Decouple APIs from command transport / IPC
- No crypto, heap, file I/O
- Dynamic loading / dlopen API

TPM Access Broker and Resource Manager (TAB/RM)
- Abstract Storage Limitations
- No crypto
- Power management

TPM Device Driver
- Device Interface (CRB / polling)
- Pre-boot log handoff

Kernel

The tpm2-software core projects

- **tpm2-tss (core library)**
  - Autotools, pkg-config, deps: libcrypto OR libgcrypt coming deps: libcurl, libjson-c
- **tpm2-abrmd (user space RM)**
  - Autotools, pkg-config, deps: libdbus, libglib
- **tpm2-tools (CLI tools)**
  - Autotools, pkg-config, deps: libcrypto, libcurl
- **CI with ~80% coverage targets, scanbuild, coverity, CII best practice, lgtm, …**
- **Building multi-distro CI using docker**

People and community

- **Maintainers:**
  - Bill, ~~Elias~~ Jonas (update), Imran, Jürgen, John, Phil, Peter, Tadeusz, and me
- **>100 contributors**
- **Packaged: Arch, Debian, Fedora, RHEL, SuSE, Ubuntu, …**
- **Further platforms: Windows, QNX, VxWorks, FreeBSD, …., Arduino**
- **https://tpm2-software.github.io**

UC: Shielded key storage and usage

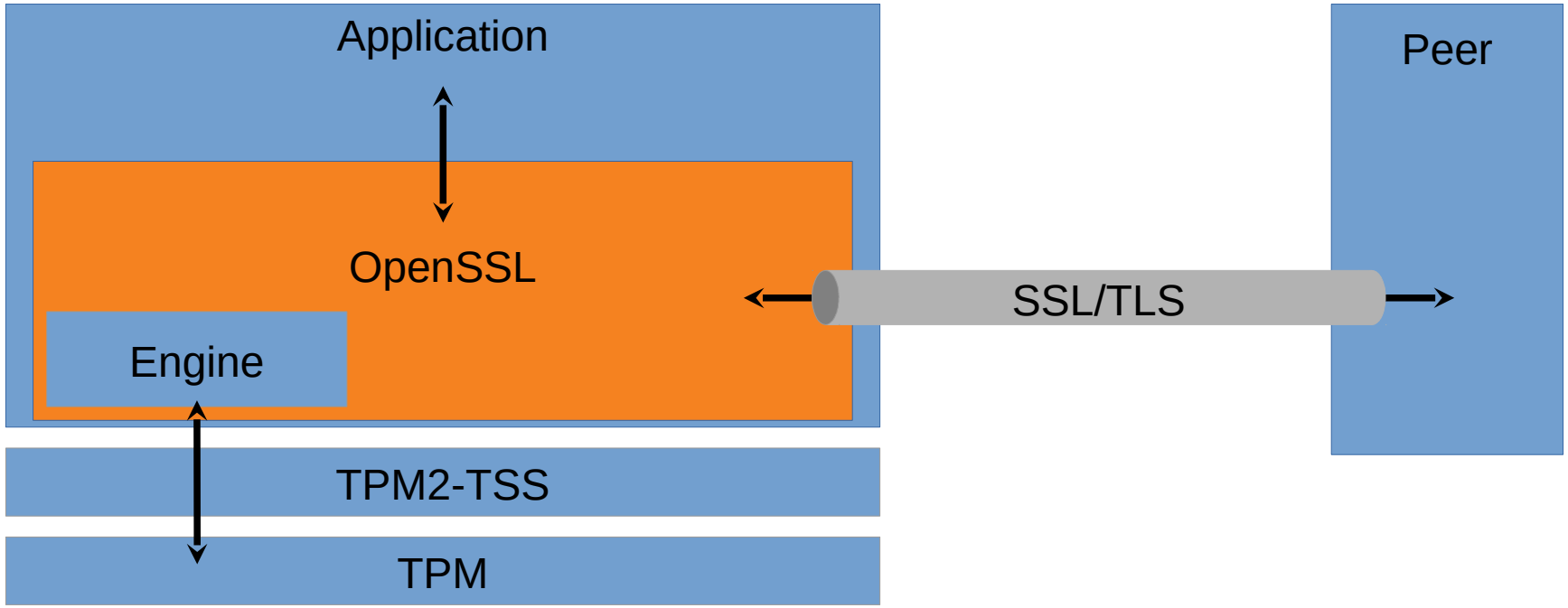
- **Keys in RAM are always dangerous**
  - “Heartbleed”
- **Keys on Disk are always dangerous**
  - You can protect them with user passwords but they can be bruteforced
  - Servers have no unlock step
  - Embedded devices have no unlock step
- **So how do you prevent ID-cloning ?**

  → Use TPM

UC: Shielded key storage and usage

- **How do you use the TPM ?**

  → easy: tpm2-tss-engine



UC: (General) user authentication

- **Typical SmartCard workflow (PKCS11)**
  - Proof of possession (of smartcard)
  - Proof of knowledge (of PIN not password)
  - More secure and convenient than passwords

  → tpm2-pkcs11 (Virtual SmartCard)

  - Proof of possession (of TPM-holding device)
  - Proof of knowledge
  - Fully compatible

UC: (VPN) user authentication

- **UserName + Password ?**

  → Machine + UserPassword !

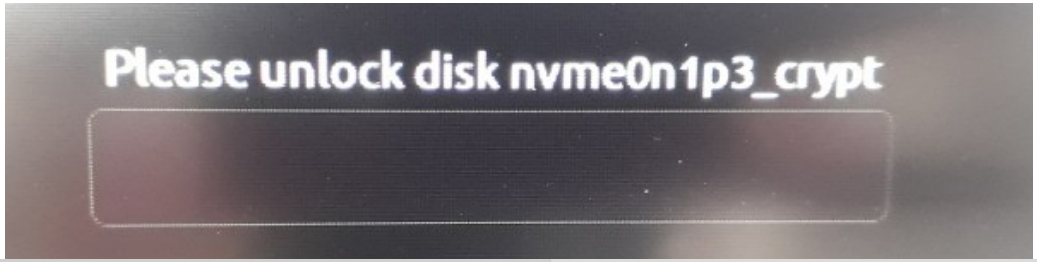
  Adding security to network access
- **OpenConnect (David Woodhouse)**
  - Reuse (copy) of tpm2-tss-engine
- **Strongswan**
  - Implements Attestation and RIMs as well
- **OpenVPN via tpm2-tss-engine ?**
- **Missing WireGuard, Tinc, ...**

UC: Disk encryption

- **“Bitlocker for Linux”**
  - Binding the disk to the machine
  - Short PIN instead of long passwords
  - No more dictionary attacks
- **Even more utility in other areas**
  - Data Center: People stealing HDDs from the rack
  - Embedded device once more
  - Binding to BIOS integrity status (local attestation)

UC: Disk encryption

- **cryptsetup(-tpm) / LUKS2**
  - Rearchitecting with Milan
  - Making cryptsetup ”module-aware”




```
{
 "keyslots": {
  "1": {
   "type": "tpm2",
   "key_size": 32,
   "area": {
    "type": "tpm2nv",
    "nvindex": 2929459
    "pcrselection": 0,
    "pcrbanks": 1,
    "noda": true
   },
```

```
afuchs@pc-fuchslap3:~/Dokumente/oss-tss/cryptsetup-tpm-incubator$ ./cryptsetup luksFormat --type=luks2 --tpm disk.img

WARNING!
========
Hiermit werden die Daten auf »disk.img« unwiderruflich überschrieben.

Are you sure? (Type uppercase yes): YES
Geben Sie die Passphrase für »disk.img« ein:
Passphrase bestätigen:
afuchs@pc-fuchslap3:~/Dokumente/oss-tss/cryptsetup-tpm-incubator$ ./cryptsetup luksOpen disk.img --test-passphrase
Geben Sie die Passphrase für »disk.img« ein:
afuchs@pc-fuchslap3:~/Dokumente/oss-tss/cryptsetup-tpm-incubator$
```

What’s missing ?

- **Attestation (see mjg’s talk)**
- **More core system integration**
  - 802.1X: NetworkManager, systemd-networkd
  - User keyrings: gnome-keyring, kwallet
  - VPNs: Wireguard, Tinc, …
  - Signing: GnuPG
  - WebCrypto / WebAuthn (Firefox, Chrome, ...)
  - ……
- **2nd maintainer for tpm2-tss-engine :-)**

So what do I want from you ?

- **Help spread TPM support to the whole core infrastructure**
  - Making the world more convenient and secure
  - Targeting: Desktop, Server, NetworkEq, Automotive, Railway, Energy, IoT, ...
- **Making Linux utilize the TPM on your platform to the max**

What do you need to do ?

- **Take an example and “just copy”**
- **Steps to take:**
  - Identify the crypto operations
  - Key storage scheme:
    - TPM persistent, TSS keystore, self managed
  - Key access control
    - Just “on the device”, password, policies
  - Implement the crypto
- **Come talk to me / Drop me a mail...**

What would this code look like ?

- **Fapi:**

```
r = Fapi_CreateKey(context, "HS/SRK/mySignKey", SIGN_TEMPLATE,
                   policy_name, PASSWORD);
r = Fapi_Sign(context, "HS/SRK/mySignKey", NULL,
              &digest.buffer[0], digest.size, &signature, &signatureSize,
              &publicKey, NULL);
```

- **Esys:**

```
r = Esys_CreatePrimary(esys_context, ESYS_TR_RH_OWNER, ESYS_TR_PASSWORD,
                                ESYS_TR_NONE, ESYS_TR_NONE,
                                &inSensitivePrimary, &inPublic, &outsideInfo, &creationPCR,
                                &primaryHandle, &outPublic, &creationData, &creationHash,
                                &creationTicket);
r = Esys_Create(esys_context, primaryHandle,
                     ESYS_TR_PASSWORD, ESYS_TR_NONE, ESYS_TR_NONE,
                     &inSensitive2, &inPublic2, &outsideInfo2, &creationPCR2,
                     &outPrivate2, &outPublic2,
                     &creationData2, &creationHash2, &creationTicket2);
r = Esys_Load(esys_context, primaryHandle, ESYS_TR_PASSWORD, ESYS_TR_NONE,
                   ESYS_TR_NONE, outPrivate2, outPublic2, &loadedKeyHandle);
r = Esys_Sign(esys_context, primaryHandle,
                   ESYS_TR_PASSWORD, ESYS_TR_NONE, ESYS_TR_NONE,
                   &pcr_digest_zero, &inScheme, &hash_validation, &signature);
```

What would this look like ?

- **tpm2-tss-engine:**

- ```
  while (num > 0) {
      r = Esys_GetRandom(eactx.ectx,
                ESYS_TR_NONE, ESYS_TR_NONE, ESYS_TR_NONE, num, &b);
      ERRchktss(rand_bytes, r, esys_auxctx_free(&eactx); goto end);
      memcpy(buf, &b->buffer, b→size);
      num -= b→size;
      buf += b→size;
      free(b);
  }
  ```

- ```
  r = init_tpm_key(&eactx, &keyHandle, tpm2Data);
  ERRchktss(rsa_priv_dec, r, goto out);
  r = Esys_RSA_Decrypt(eactx.ectx, keyHandle,
                    ESYS_TR_PASSWORD, ESYS_TR_NONE, ESYS_TR_NONE,
                    &cipher, &inScheme, &label, &message);
  ERRchktss(rsa_priv_dec, r, goto out);
  ```

Questions ?

https://tpm2-software.github.io