TPM2 Software Community

https://github.com/tpm2-software

Philip Tricca (Intel)
Andreas Fuchs (Fraunhofer SIT)

Agenda

- Intro & Architecture
- boot: tcti-uefi
- verify system: tpm2-totp
- decrypt disk: cryptsetup/clevis
- vpn: strongswan / openconnect
- server: openssl
- learning, experimenting, prototyping
- develop: Join us

TSS2 Design

- Use-case driven
  - Support for constrained environments to full OS: Layered design
  - Separate transport layer from APIs
  - Both synchronous and async: event-driven programming
  - Details exposed if needed, “sane defaults” otherwise
- Lower layers provide data transport & direct access to TPM2 commands
  - “Expert” applications in constrained environments
  - Minimal dependencies (c99, libc)
- Upper layers provide convenience functions & abstractions
  - Crypto for sessions, dynamic memory allocation, transport layer configuration
  - More features → more dependencies

TSS2 Design

System API (tss2-sys)
- 1:1 to TPM2 cmds
- Command / Response serialization
- No file I/O
- No crypto
- No heap / malloc

Enhanced SYS (tss2-esys)
- Automate crypto for HMAC / encrypted sessions
- Dynamic TCTI loading
- Memory allocations
- No file I/O

Feature API (FAPI)
- Spec in draft form
- No implementation yet
- File I/O
- Requires heap
- Automate retries
- Context based state
- Must support static linking

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

User Space

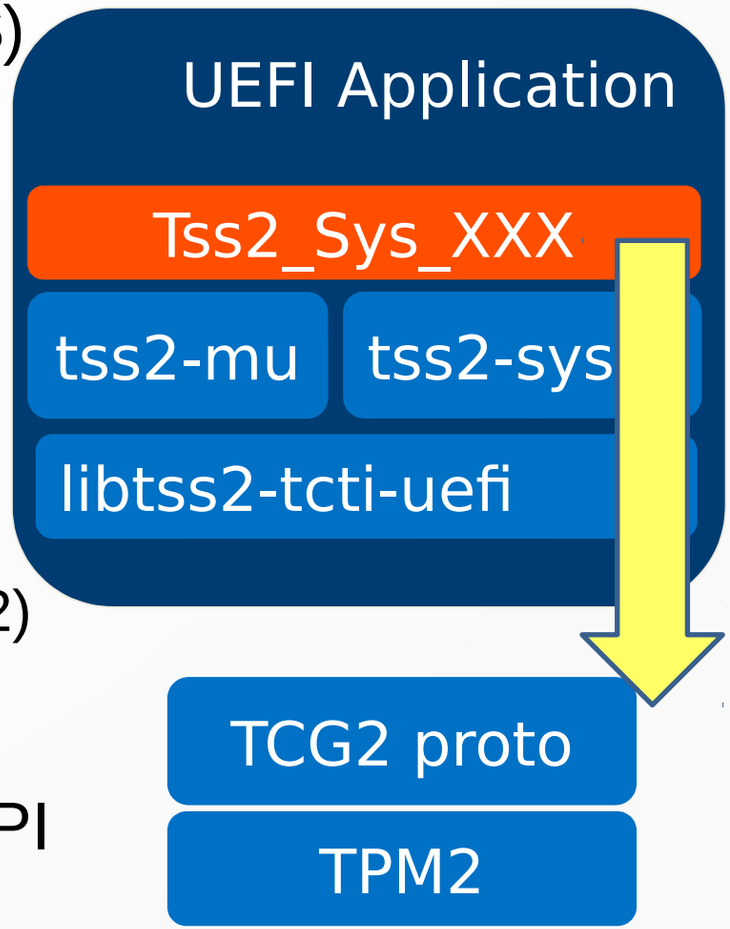
Kernel

Use-Cases

- Frame use-cases in PC bootflow: “a day in the life”
- Firmware / Bootloader
- Early-boot (initramfs)
- Full userspace
  - Network Connectivity / Authentication
  - Prototyping / Development
  - Debug / Test

Boot / Firmware

- UEFI provides minimal support (NV storage == $$)
  - Query UEFI protocol & PCR bank settings
  - HashLogExtendEvent: measure stuff
  - SendCommand: send command buffer
- Use-case: UEFI applications & bootloaders
  - System manufacturing & provisioning & mgmt
  - Encrypted boot partition with TPM protected keys (grub2)
- TCTI built on TCG2 UEFI protocol: libtss2-tcti-uefi
- Enable use of all TPM2 commands via tss2-sys API
  - https://github.com/tpm2-software/tpm2-tcti-uefi
  - https://firmware.intel.com/content/tour-beyond-bios-uefi-tpm2-support-edkii



swtpm + Qemu + OVMF

- Output from GetCapability TPM2 command
- TPM2_PT_NV_BUFFER_MAX is 0x400 → 1k
- Gnu-efi version > 3.0.8

```
TPM2_PT_MAX_SESSION_CONTEXT:
  value: 0x00000194
TPM2_PT_PS_FAMILY_INDICATOR:
  value: 0x322E3000
TPM2_PT_PS_LEVEL:
  value: 0x00000000
TPM2_PT_PS_REVISION:
  value: 0x00000096
TPM2_PT_PS_DAY_OF_YEAR:
  value: 0x00000106
TPM2_PT_PS_YEAR:
  value: 0x000007E2
TPM2_PT_SPLIT_MAX:
  value: 0x00000080
TPM2_PT_TOTAL_COMMANDS:
  value: 0x0000006D
TPM2_PT_LIBRARY_COMMANDS:
  value: 0x0000006D
TPM2_PT_VENDOR_COMMANDS:
  value: 0x00000000
TPM2_PT_NV_BUFFER_MAX:
  value: 0x00000400
TPM2_PT_MODES:
  raw: 0x00000000
FS0:\> _
```

swtpm + Qemu + OVMF

- Versions < 3.0.8 are missing ‘memcpy’ & ‘memset’

```
FS0:\> tpm2-get-caps-fixed.efi
LibLocateProtocol returned handle to EFI_TCG2_PROTOCOL: 0x7AB6C00
LibLocateProtocol returned handle to EFI_TCG2_PROTOCOL: 0x7AB6C00
!!!! X64 Exception Type - 06(#UD - Invalid Opcode)  CPU Apic ID - 00000000 !!!!
RIP  - 00000000065B7024, CS  - 0000000000000038, RFLAGS - 0000000000000286
RAX  - 00000000065A16EF, RCX - 0000000007F13380, RDX - 0000000000000408
RBX  - 0000000000000000, RSP - 0000000007F12E70, RBP - 0000000007F132C0
RSI  - 0000000000000000, RDI - 0000000007F13380
R8   - 0000000000000004, R9  - 000000000671B065, R10 - 0000000000001000
R11  - 0000000000000020, R12 - 0000000007F13888, R13 - 0000000007F13898
R14  - 0000000007F13900, R15 - 0000000000000000
DS   - 0000000000000030, ES  - 0000000000000030, FS  - 0000000000000030
GS   - 0000000000000030, SS  - 0000000000000030
CR0  - 0000000080010033, CR2 - 0000000000000000, CR3 - 0000000007C01000
CR4  - 0000000000000668, CR8 - 0000000000000000
DR0  - 0000000000000000, DR1 - 0000000000000000, DR2 - 0000000000000000
DR3  - 0000000000000000, DR6 - 00000000FFFF0FF0, DR7 - 0000000000000400
GDTR - 0000000007BEEA98 0000000000000047, LDTR - 0000000000000000
IDTR - 00000000076A8018 0000000000000FFF,   TR - 0000000000000000
FXSAVE_STATE - 0000000007F12AD0
!!!! Find image based on IP(0x65B7024) (No PDB)  (ImageBase=0000000006594000, EntryPoint=00000000065B1054) !!!!
```

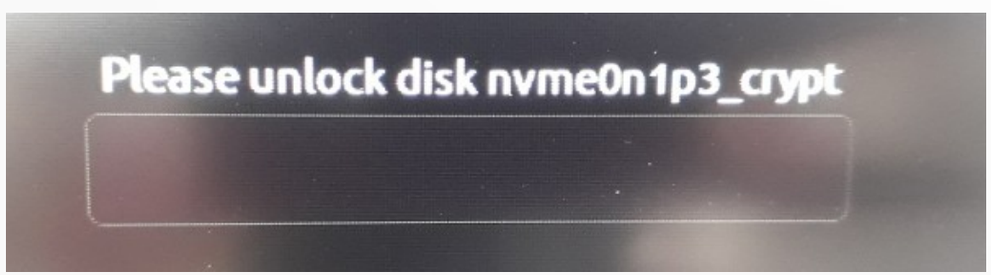
Verify the system: tpm2-totp

- Based on tpm-totp by Matthew Garret
- Reimplementation for TPM2 (using ESYS)
  - Built as a library for re-use + CLI tool
  - Uses TPM2 features (HMAC) for additional security
- The challenges:
  - Are you entering your password into your PC ?
  - Was you BIOS/Kernel/Initrd altered by an evil maid ?
- The solution:
  - Time-based One-Time Passwords to authenticate your PC to you
  - Verification using e.g. you cell phone
- https://github.com/tpm2-software/tpm2-totp


afuchs@pc-fuchslap3: ~/Dokumente/oss-tss/tpm2-totp
Datei Bearbeiten Ansicht Suchen Terminal Reiter Hilfe
afuchs@pc-fuchslap3: ~/Dokumente/oss-tss/tp...
afuchs@pc-fuchslap3: ~/Dokumente/oss-tss/cry...
afuchs@pc-fuchslap3: ~/Dokumente/oss-tss/tp...
afuchs@pc-fuchslap3:~/Dokumente/oss-tss/tpm2-totp$ ./tpm2-totp generate -p 0,2,4,6
Calling Esys_GetRandom for 20 bytes
Calling Esys_CreatePrimary
otpauth://totp/TPM2-TOTP?secret=UJCTACMSADJFX545VP44YBNP5HEMXIVE
afuchs@pc-fuchslap3:~/Dokumente/oss-tss/tpm2-totp$ ./tpm2-totp calculate -t; echo
2019-02-02 13:33:24: 612934
afuchs@pc-fuchslap3:~/Dokumente/oss-tss/tpm2-totp$
DEMO

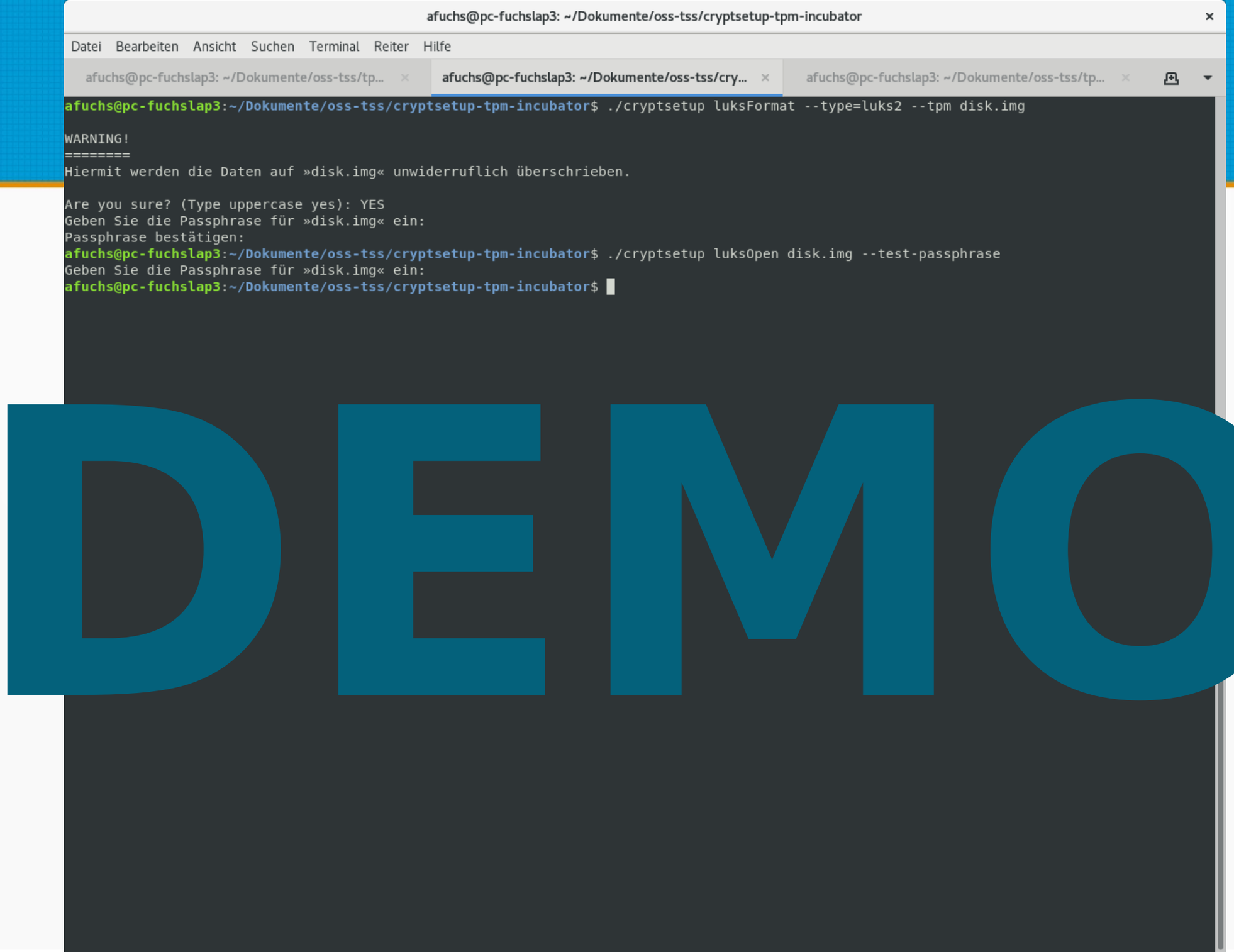
Decrypting disk: cryptsetup

- https://gitlab.com/cryptsetup/cryptsetup/merge_request/51
- UseCase: Bitlocker (from Windows)
  - Use dictionary attack protection (Do you prefer a PIN ?)
  - Actually „change“ and not „add“ a password
  - PR#51 in cryptsetup GitLab




```
{
  "keyslots": {
    "0": {
      "type": "luks2",
      "key_size": 32,
      "kdf": {
        ...
      },
      "af": {
        "type": "luks1",
        "hash": "sha256",
        "stripes": 4000
      },
      "area": {
        "type": "raw",
        "encryption": "aes-xts-plain64",
        "key_size": 32,
        "offset": "32768",
        "size": "131072"
      }
    }
```

```
{
  "keyslots": {
    "1": {
      "type": "tpm2",
      "key_size": 32,
      "area": {
        "type": "tpm2nv",
        "nvindex": 29294593,
        "pcrselection": 0,
        "pcrbanks": 1,
        "noda": true
    },
```

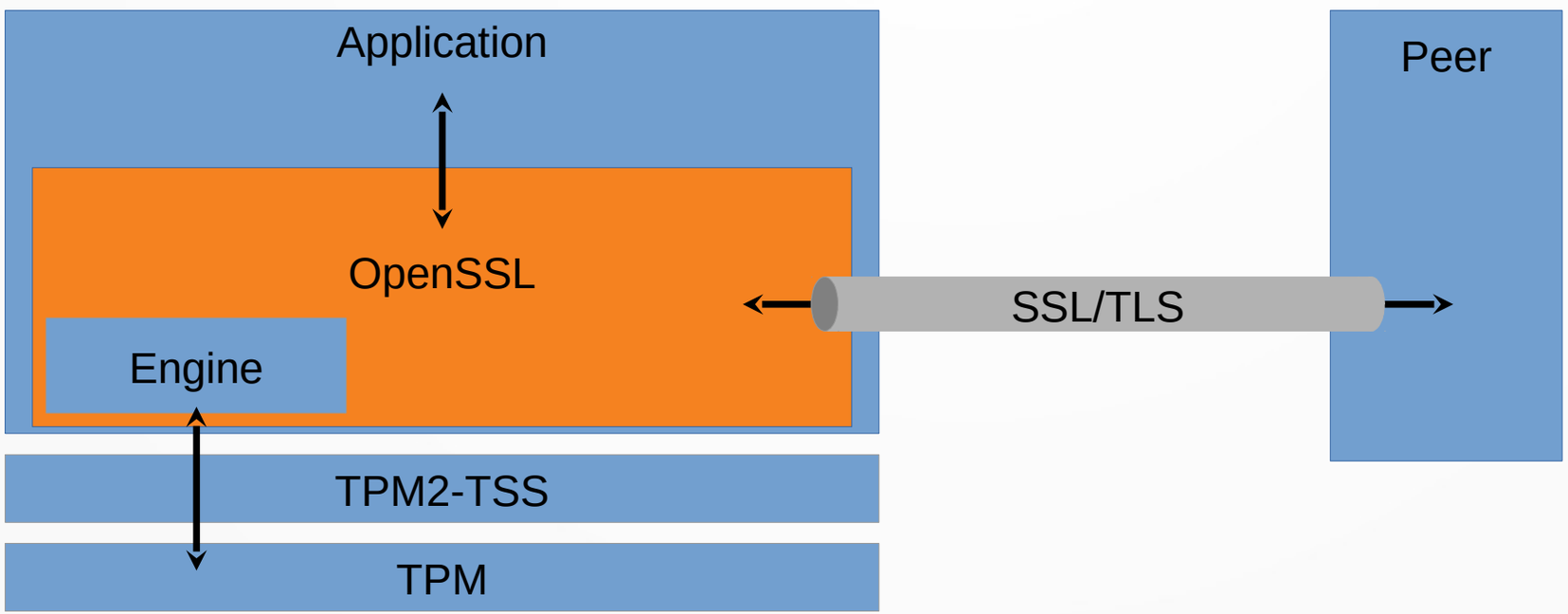

afuchs@pc-fuchslap3: ~/Dokumente/oss-tss/cryptsetup-tpm-incubator
Datei Bearbeiten Ansicht Suchen Terminal Reiter Hilfe
afuchs@pc-fuchslap3: ~/Dokumente/oss-tss/tp...
afuchs@pc-fuchslap3: ~/Dokumente/oss-tss/cry...
afuchs@pc-fuchslap3: ~/Dokumente/oss-tss/tp...
afuchs@pc-fuchslap3:~/Dokumente/oss-tss/cryptsetup-tpm-incubator$ ./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$
DEMO

Decrypting disks: Clevis

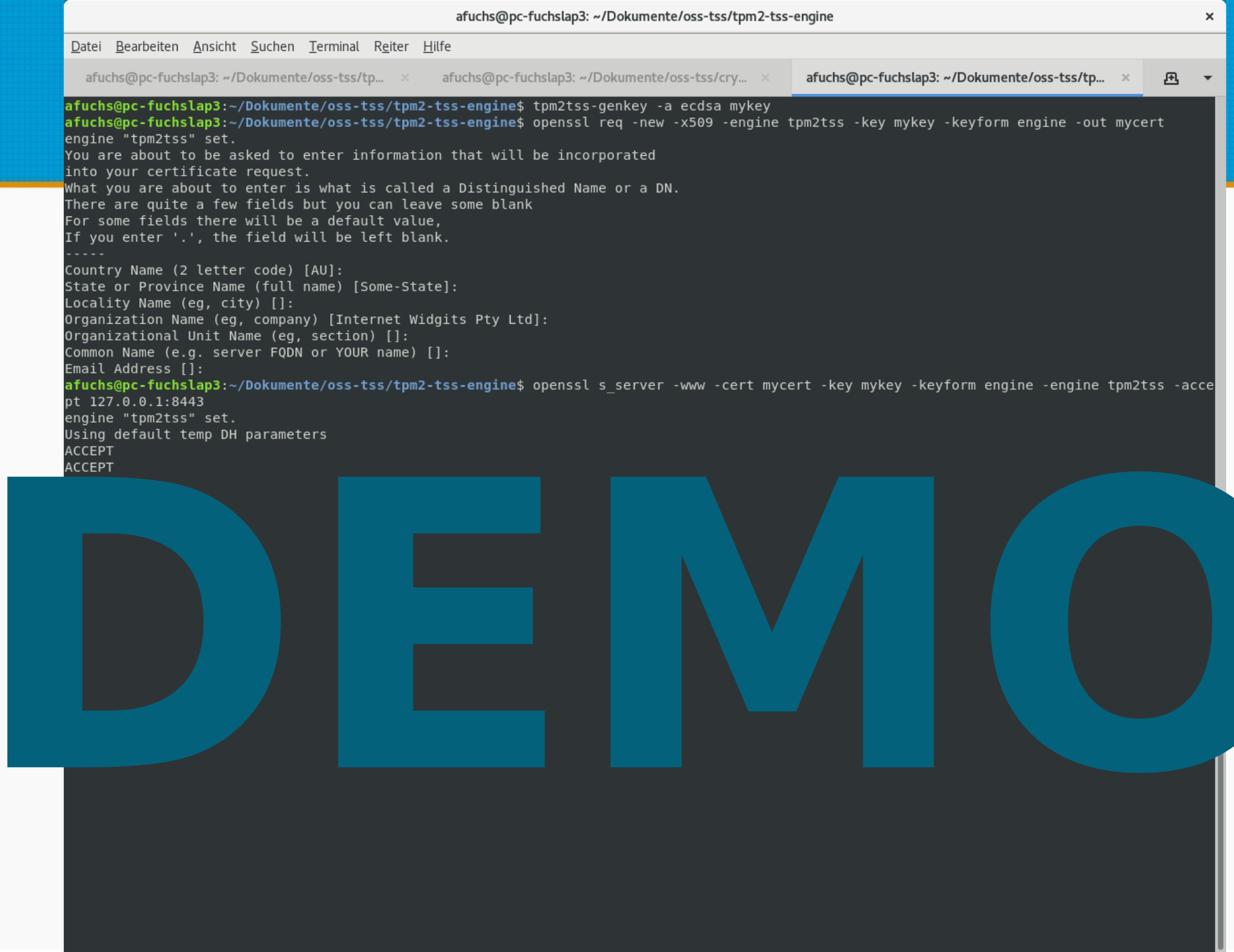
- https://en.wikipedia.org/wiki/Clevis_fastener
- Plugable framework for automating decryption
- Plugins called “pins”, TPM2 pin implemented with TSS2
  - Automated decryption of data
  - Automated unlocking of LUKS volumes
  - Implementation based on tpm2-tools
- Javier Martinez Canillas talk @ RedHat devconf 2019
  - https://devconfcz2019.sched.com/event/Jcir/applications-of-tpm-20
  - Javier maintains https://github.com/tpm2-software/tpm2-tools
- “Red Hat development efforts based on the TCG specifications”

Server Connection: tpm2-tss-engine

- An OpenSSL engine
- https://github.com/tpm2-software/tpm2-tss-engine



- export OPENSSL_CONF=/path/to/openssl-tpm.cnf


afuchs@pc-fuchslap3: ~/Dokumente/oss-tss/tpm2-tss-engine
Datei Bearbeiten Ansicht Suchen Terminal Reiter Hilfe
afuchs@pc-fuchslap3: ~/Dokumente/oss-tss/tp...
afuchs@pc-fuchslap3: ~/Dokumente/oss-tss/cry...
afuchs@pc-fuchslap3: ~/Dokumente/oss-tss/tp...
afuchs@pc-fuchslap3:~/Dokumente/oss-tss/tpm2-tss-engine$ tpm2tss-genkey -a ecdsa mykey
afuchs@pc-fuchslap3:~/Dokumente/oss-tss/tpm2-tss-engine$ openssl req -new -x509 -engine tpm2tss -key mykey -keyform engine -out mycert
engine "tpm2tss" set.
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.

Country Name (2 letter code) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:
afuchs@pc-fuchslap3:~/Dokumente/oss-tss/tpm2-tss-engine$ openssl s_server -www -cert mycert -key mykey -keyform engine -engine tpm2tss -acce
pt 127.0.0.1:8443
engine "tpm2tss" set.
Using default temp DH parameters
ACCEPT
ACCEPT

Authentication: VPN Client / Server

- Multiple VPN clients supporting TSS2
- StrongSwan (5.7.0+)
  - TPM2 support via TSS2 for both client and server
  - Implementation of TNC algorithms
  - https://wiki.strongswan.org/projects/strongswan/wiki/TpmPlugin
- OpenConnect
  - SSL VPN client, drop-in replacement for Cisco SSL client
  - Uses OpenSSL engine
  - http://www.infradead.org/openconnect/tpm.html

Authentication: PKCS#11

- Cryptographic token API: smartcards, usb tokens
- Wide adoption for authentication: PAM, SSH, NSS, OpenSSL
- Use PKCS#11 as compatibility layer to TPM2
  - https://github.com/tpm2-software/tpm2-pkcs11
  - Author / Maintainer William Roberts
- Featured in ‘AWS re:Invent’ IoT Greengrass demo
  - Use hardware security mechanism / TPM2 to protect secrets
  - https://youtu.be/l0tmTaIqAK8?t=1703
- PKCS#11 for SSH auth
  - https://github.com/tpm2-software/tpm2-pkcs11/blob/master/docs/INITIALIZING.md
  - https://github.com/tpm2-software/tpm2-pkcs11/blob/master/docs/SSH.md

tpm2-pkcs11 for SSH auth

- Initialize metadata store for object mapping
  - Default $HOME/.tpm2_pkcs11
  - tpm2_ptool.py init --pobj-pin=mypobjpin
  - tpm2_ptool.py addtoken --pid=1 --pobj-pin=mypobjpin --sopin=mysopin \ --userpin=myuserpin --label=mylabel
  - tpm2_ptool.py addkey --algorithm=ecc384 --label=mylabel –userpin=myuserpin
- Use for ssh auth
  - ssh-keygen -D ./src/.libs/libtpm2_pkcs11.so | tee my.pub
  - Insert into authrized_keys file on remote host
  - ssh -I /usr/local/lib/libtpm2_pkcs11.so example.com

tpm2-tools

- Command line tools automating TPM2 operations
  - https://github.com/tpm2-software/tpm2-tools
  - Often a user's first experience with TPM2 / TSS2
- Much refactoring in 2017 & 2018
  - Started as a clone of the IBM command line tools from TSS for TPM 1.2
  - Initial implementation pre-dates availability of libtss2-esys
  - Has evolved to a near 1:1 mapping to TPM2 commands
  - Maintainership shared between RedHat & Intel
- Now using libtss2-esys / ESAPI
- Individual tool execs can be scripted to achive higher level task

tpm2-tool example

- Updated example from Davide Guerri @ FOSDEM 2017
- Sign data with TPM2 key / verify signature with OpenSSL
  - tpm2_createprimary --hierarchy o --out-context pri.ctx
  - tpm2_create --context-parent pri.ctx --pubfile sub.pub --privfile sub.priv
  - tpm2_load --context-parent file:sub.priv --pubfile sub.pub --privfile sub.priv --out-context sub.ctx
  - openssl dgst -sha1 -binary -out hash.bin msg.txt
  - tpm2_sign --key-context file:sub.ctx --format plain --digest hash.bin --sig hash.plain
  - tpm2_readpublic --key-context file:sub.ctx --format der --out-file sub-pub.der
  - openssl dgst -verify sub-pub.der -keyform der -sha1 -signature hash.plain msg.txt

language support

- Google tpm-js: https://google.github.io/tpm-js
- python bindings: work in progress
- OO-wrapper around python CFFI bindings
- https://github.com/tpm2-software/tpm2-tss/pull/1248

```
from pytpm2tss import *

e = EsysContext()
e.Startup(TPM2_SU.CLEAR)
random_bytes = e.GetRandom(5)
```

bindings: pytpm2tss

```
inSensitive = TPM2B_SENSITIVE_CREATE()
inPublic = TPM2B_PUBLIC()
outsideInfo = TPM2B_DATA()
creationPCR = TPML_PCR_SELECTION()

inPublic.publicArea.type = TPM2_ALG.ECC
inPublic.publicArea.nameAlg = TPM2_ALG.SHA1
inPublic.publicArea.objectAttributes = TPMA_OBJECT.USERWITHAUTH | \
                            TPMA_OBJECT.SIGN_ENCRYPT | TPMA_OBJECT.RESTRICTED | \
                            TPMA_OBJECT.FIXEDTPM | TPMA_OBJECT.FIXEDPARENT | \
                            TPMA_OBJECT.SENSITIVEDATAORIGIN
inPublic.publicArea.parameters.eccDetail.scheme.scheme = TPM2_ALG.ECDSA
inPublic.publicArea.parameters.eccDetail.scheme.details.ecdsa.hashAlg = TPM2_ALG.SHA256
inPublic.publicArea.parameters.eccDetail.symmetric.algorithm = TPM2_ALG.NULL
inPublic.publicArea.parameters.eccDetail.kdf.scheme = TPM2_ALG.NULL
inPublic.publicArea.parameters.eccDetail.curveID = TPM2_ECC.NIST_P256



x, _, _, _, _ = e.CreatePrimary(e.tr.OWNER,
                    inSensitive, inPublic, outsideInfo, creationPCR,
                    session1=e.tr.PASSWORD)

signature = e.Sign(x, digest, scheme, None, session1=e.tr.PASSWORD)
```

Debugging: Wireshark

- Not enough debugging / teaching tools (none?)
- TPM2 buffers can be a lot like network packets
  - Communicating with TPM2 simulator over loopback
  - Command / response stream packed network byte order
- Patches in 2.9 development tree thanks to Tadeusz Struk
  - https://1.na.dl.wireshark.org/src/wireshark-2.9.0.tar.gz
- Current state: new
  - Listen on loopback, sniff traffic between application & simulator
  - UI integration, parsing 3 of ~120 TPM2 commands

Wireshark: headers

Source	Destination	Protocol	Length	Info
127.0.0.1	127.0.0.1	TPM	1172	59374 → 2321, [TPM Request], Command TPM2_CC_EncryptDecrypt, len(1106)
127.0.0.1	127.0.0.1	TPM	75	59374 → 2321, [TPM Request], Platform Command TPM_SEND_COMMAND, len(9)
127.0.0.1	127.0.0.1	TPM	80	2321 → 59374, [TPM Response], Response Code TPM2_RC_RETRY - the TPM was not able to start the command, len(14)
127.0.0.1	127.0.0.1	TPM	70	2321 → 59374, [TPM Response], Response size 10, len(4)
127.0.0.1	127.0.0.1	TPM	1172	59374 → 2321, [TPM Request], Command TPM2_CC_EncryptDecrypt, len(1106)
127.0.0.1	127.0.0.1	TPM	75	59374 → 2321, [TPM Request], Platform Command TPM_SEND_COMMAND, len(9)
127.0.0.1	127.0.0.1	TPM	215	2321 → 59374, [TPM Response], Response Code TPM2 Success, len(149)
127.0.0.1	127.0.0.1	TPM	70	2321 → 59374, [TPM Response], Response size 145, len(4)
127.0.0.1	127.0.0.1	TPM	183	59374 → 2321, [TPM Request], Command TPM2_CC_CreateLoaded, len(117)
127.0.0.1	127.0.0.1	TPM	75	59374 → 2321, [TPM Request], Platform Command TPM_SEND_COMMAND, len(9)
127.0.0.1	127.0.0.1	TPM	89	2321 → 59374, [TPM Response], Response Code TPM2 Success, len(23)
127.0.0.1	127.0.0.1	TPM	70	2321 → 59374, [TPM Response], Response size 19, len(4)
127.0.0.1	127.0.0.1	TPM	101	59374 → 2321, [TPM Request], Command TPM2_CC_PolicyAuthorizeNV, len(35)
127.0.0.1	127.0.0.1	TPM	75	59374 → 2321, [TPM Request], Platform Command TPM_SEND_COMMAND, len(9)
127.0.0.1	127.0.0.1	TPM	89	2321 → 59374, [TPM Response], Response Code TPM2 Success, len(23)
127.0.0.1	127.0.0.1	TPM	70	2321 → 59374, [TPM Response], Response size 19, len(4)
127.0.0.1	127.0.0.1	TPM	135	59374 → 2321, [TPM Request], Command TPM2_CC_NV_Write, len(69)

- Sniff traffic between application & simulator
- Layer 2 & 3 headers + 10 byte TPM2 header
- Parse & display TPM2 command / response header

Wireshark: command body

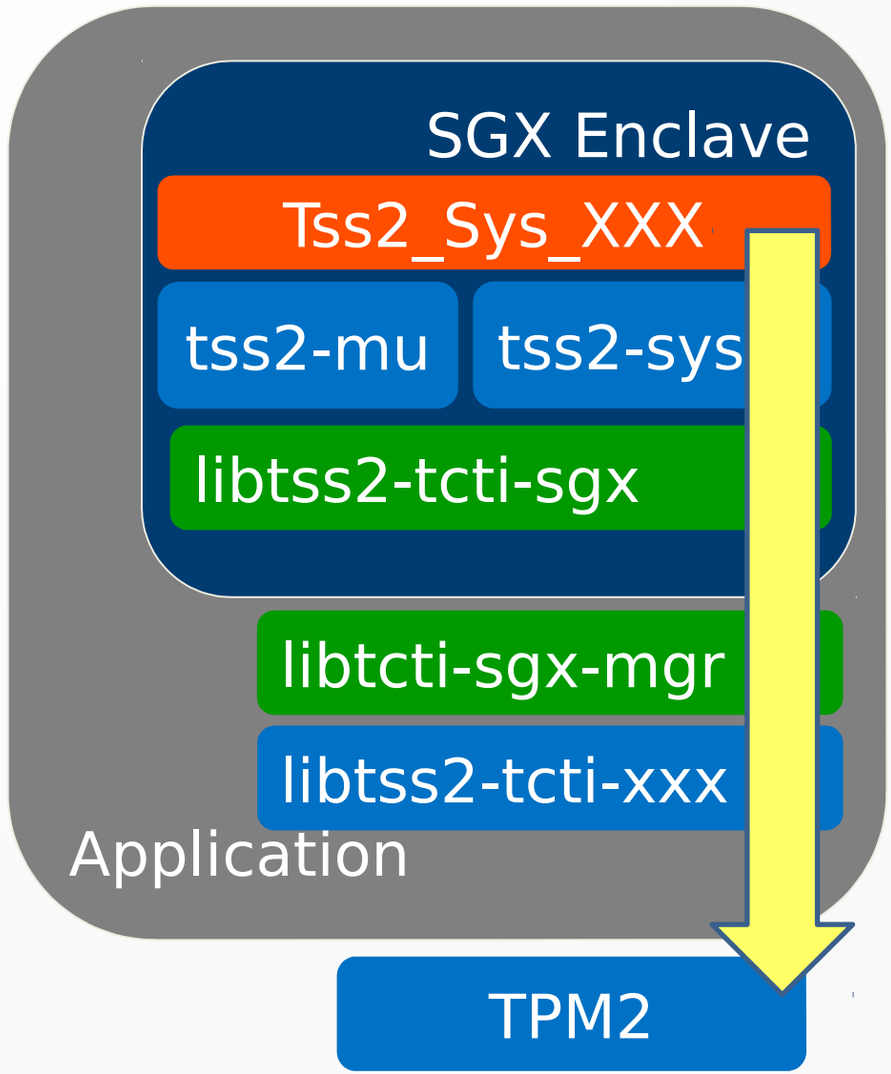
```
▸ Frame 227: 183 bytes on wire (1464 bits), 183 bytes captured (1464 bits) on interface 0
▸ Ethernet II, Src: 00:00:00:00:00:00 (00:00:00:00:00:00), Dst: 00:00:00:00:00:00 (00:00:00:00:00:00)
▸ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▸ Transmission Control Protocol, Src Port: 59374, Dst Port: 2321, Seq: 307, Ack: 156, Len: 117
▾ TPM2.0 Protocol
  ▾ TPM2.0 Header, TPM2_CC_CreateLoaded
      Tag: Command with authorization Sessions (0x8002)
      Command size: 117
      Command Code: TPM2_CC_CreateLoaded (0x00000191)
  ▾ Handle Area
      TPMI_RH_HIERARCHY: TPM2_RH_OWNER (0x40000001)
  ▾ Authorization Area
      AUTHAREA SIZE: 41
      TPMI_SH_AUTH_SESSION: TPM2_RS_PW (0x40000009)
      AUTH NONCE SIZE: 0
      AUTH NONCE: <none>
    ▾ Session attributes
        .... ...0 = SESSION_CONTINUESESSION: Not set
        .... ..0. = SESSION_AUDITEXCLUSIVE: Not set
        .... .0.. = SESSION_AUDITRESET: Not set
        ...0 0... = SESSION_RESERVED: Not set
        ..0. .... = SESSION_DECRYPT: Not set
        .0.. .... = SESSION_ENCRYPT: Not set
        0... .... = SESSION_AUDIT: Not set
      SESSION AUTH SIZE: 32
      SESSION AUTH: 0000000000000000000000000000000000000000000000000000...
  TPM SENSITIVE CREATE SIZE: 4
  TPM SENSITIVE CREATE: 00000000
  TPM TEMPLATE SIZE: 50
  TPM TEMPLATE: 0025000b000600720020063a24dc2fc932c3b8a085ca6727...
```

TEE Enabling: tpm2-tcti-sgx

- Trusted Execution Environment (TEE)
  - stronger separation guarantees than provided by OS processes
  - SmartCard / TPM achieve similar goal, requires new hardware
  - CPUs being developed to support TEEs
- TEEs are islands
  - Great so long as you have everything you need with you
  - Interacting with external entities (TPM2) has value
  - Doing so adds complexity & risk
  - This is interesting and worth a **prototype**

TEE Enabling: tpm2-tcti-sgx

- Transport library: send command / response across enclave boundary
- Mgmt library: broker command / response between enclave & TSS2
- Good way to explore / learn
  - Build process for SGX enclaves & supporting libraries
  - Testing strategies
  - Implications of crossing domain boundary
  - Mitigation / security association between enclave & TPM2
- https://github.com/flihp/tpm2-tcti-sgx



References

- tpm2-tcti-uefi: https://github.com/tpm2-software/tpm2-tcti-uef
- tpm2-totp: https://github.com/tpm2-software/tpm2-totp
- cryptsetup (TPM2): gitlab.com/cryptsetup/cryptsetup/merge_request/51
- Clevis: https://github.com/latchset/clevis
- tpm2-pkcs11: https://github.com/tpm2-software/tpm2-pkcs11
- Strongswan: https://wiki.strongswan.org/projects/strongswan/wiki/TpmPlugin
- openconnect: http://www.infradead.org/openconnect/tpm.html
- TPM2 OpenSSL Engine: https://github.com/tpm2-software/tpm2-tss-engine
- tpm2-tools: https://github.com/tpm2-software/tpm2-tools
- pytpm2tss: https://github.com/tpm2-software/tpm2-tss/pull/1248

Develop: Join us

- OpenVPN
- WireGuard
- Tinc
- NetworkManager/wpa_supplicant 802.1X
- gnome-keyring
- KDE wallet
- GNU-TLS
- WebCrypto (Firefox, WebKit, Chromium)
- OpenSSH: PKCS#11 & ssh-agent/ gnome-keyring, native TSS2

- Talk to us: HERE & https://lists.01.org/tpm2